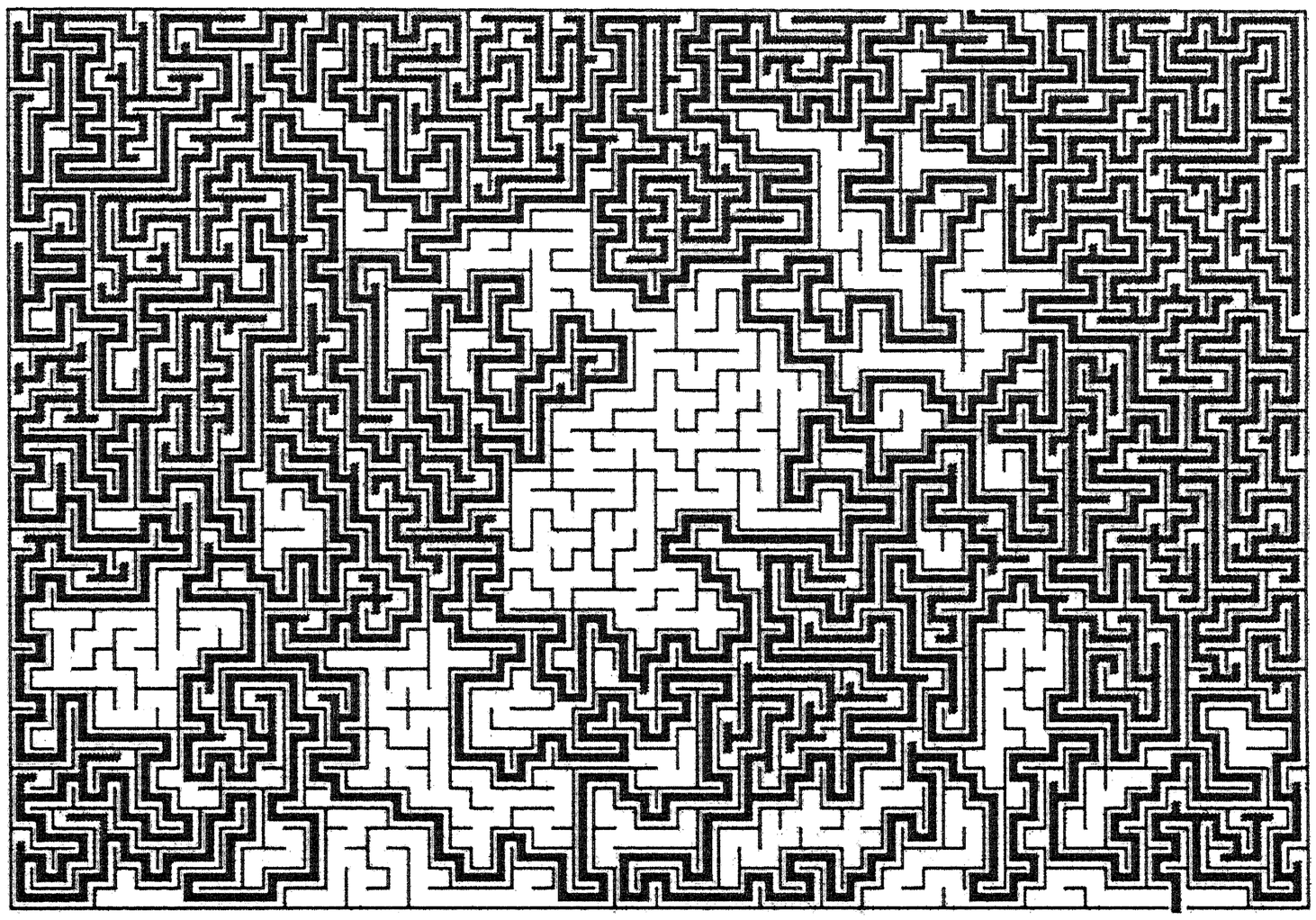
JVA 001

A-24019 *
==========CWI
Bibliotheek
Centrum voor
Wiskunde &
Informatica
=Amsterdam=

Applications of Shortest Path Methods



J.C. de Pina

CWI BIBLIOTHEEK
3 0054 00047 5351

Applications of Shortest Path Methods

Applications of Shortest Path Methods

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam,
op gezag van de Rector Magnificus
prof.dr. P.W.M. de Meijer
ten overstaan van een door het college van dekanen ingestelde
commissie in het openbaar te verdedigen in de Aula der Universiteit
op dinsdag 19 december 1995 te 13.30 uur

door José Coelho de Pina Junior

geboren te São Paulo (Brazilië)

Faculteit der Economische Wetenschappen & Econometrie

Promotor: prof.dr. A. Schrijver Universiteit van Amsterdam

Leden van de promotiecommissie:

prof.dr. N.M. van Dijk	Universiteit van Amsterdam
prof.dr. P. van Emde Boas	Universiteit van Amsterdam
dr.ir. A.M.H. Gerards	Centrum voor Wiskunde en Informatica, Amsterdam
dr. M. Laurent	Ecole Normale Supérieure, Paris
prof.dr. J.K. Lenstra	Technische Universiteit Eindhoven

Dit onderzoek werd uitgevoerd op het Centrum voor Wiskunde en Informatica (CWI) te Amsterdam met financiële steun van Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Universidade de São Paulo, de Universiteit van Amsterdam en het CWI.

Acknowledgements

Here I wish to express my gratitude to several people for their support during the preparation of this work.

First and foremost, I thank my supervisor Lex Schrijver. It was his constant support that made it possible for me to live the stimulating atmosphere of CWI during my stay. In the last 5 years, I learnt a great deal from my supervisor's deep and enlightening work. I have especially come to admire his talent for identifying the essence of a piece of mathematics, which often leads him to "proofs from the book."

Next I would like to thank Bert Gerards, who, since my first days at CWI, was always most keen to help me. In numerous occasions I benefited from his insightful remarks and explanations. The clarity of his expositions still amazes me. I am also indebted to Bert for reading this whole thesis and making a great number of valuable suggestions.

I also learnt a lot from my colleagues Maurits de Graaf and Hein van der Holst. I enjoyed our very many mathematical conversations.

Settling in Amsterdam was a pleasant experience thanks to Lex, Bert, Maurits, and Bruce Shepherd. I moved in a flat in the Bijlmermeer with their help. It was then that we learnt that the cycle lanes in the Bijlmer are 4 metres wide: during my move, we carried on two of our bikes, side by side, a rolled 4×15-metre carpet under a rather heavy rain. The carpet fit exactly on the lane—perhaps the width of the Dutch carpets take into account the width of the Bijlmer cycle lanes.

During all these years I have had the pleasure of living in a most friendly environment at CWI and, in particular, of having as officemates Han Hoogeveen, Bart Veltman, Chun Ko, Marie Collete, Hein van der Holst and Jacqueline van den Hof, and of having as "neighbours" Sem Borst and Marco Combé. I enjoyed our coffee breaks, which often turned into free Dutch lessons.

The timetabling of the Dutch Railways was the subject of other lessons I have had. Adri Steenbeek very kindly taught me about his work with Lex on this topic.

Finally, I would like to thank the very supportive staff at CWI. I am especially grateful to the library staff, who look after an excellent library.

Contents

Introduction 5

Preliminaries 8

1. General preliminaries 8
2. Graph theory 9
3. Computational complexity 12
4. The shortest path problem 16

1. Aircraft assignment and disjoint paths in acyclic digraphs 20

1.1. Introduction 20
1.2. Disjoint paths in acyclic digraphs 23
1.3. Further applications: work force scheduling 27

2. VLSI-routing and disjoint paths in planar graphs 29

2.1. Introduction 29
2.2. Some definitions and assumptions 32
2.3. Constructing the system $Ax \leq b$ 33
2.4. Shifting $Q_1, \ldots, Q_k$ 37

3. Integral potentials 39

3.1. Introduction 39

3.2. The digraph associated with $Ax \leq b$ 40

3.3. Finding a half-integral solution of $Ax \leq b$ 41

3.4. An algorithm to find a $\{0,1\}$-solution of $Cy \leq d$ 43

3.5. Rounding a half-integral solution 46

3.6. Further notes: 2-satisfiability and bidirected graphs 47

4. Optimization of integral potentials **55**

4.1. Introduction 55

4.2. Finding a maximal element 56

4.3. Finding an integral maximal element 58

5. Dynamic programming and shortest disjoint paths in planar graphs **60**

5.1. Introduction 60

5.2. Minimizing sum of lengths 61

5.3. Minimizing maximum length 65

5.4. $\mathcal{NP}$-completeness 66

5.5. Auxiliary problem 69

5.6. Case $k = 2$ 70

5.7. Case $k = 3$ 73

6. Train timetabling and bases of circuits **82**

6.1. Introduction 82

6.2. The constraints 83

6.3. The model—the periodic event scheduling problem 85

6.4. A necessary condition to feasibility 86

6.5. A backtracking algorithm 88

7. Shortest bases of circuits **92**

7.1. Introduction 92

7.2. A polynomial-time algorithm 93

7.3. Bases of circuits and signed graphs 96

7.4. Properties of circuits in a shortest circuit basis—Horton's algorithm 100

7.5. Bases of circuits and polyhedral theory 101

7.6. An application: cycle analysis of electrical networks 114

7.7. Further notes 117

8. Shortest bases of cuts **123**

8.1. Introduction 123

8.2. A polynomial-time algorithm 124

8.3. Bases of cuts and signed graphs 126

8.4. Shortest bases of cuts and the all-pairs minimum cut problem 127

8.5. Further notes: shortest cut and circuit bases in planar graphs 133

9. Shortest bases of circuits in binary matroids **135**

9.1. Introduction 135

9.2. Preliminaries on binary matroids and an optimality criterion . 136

9.3. Shortest bases of circuits in matroids with no S_8-minor 139

9.4. Shortest bases of circuits, shortest odd circuits and shortest f-circuits 143

9.5. Further notes: shortest bases of circuits in graphs with no $K_{3,3}$-minor 144

References **150**

Index **159**

Samenvatting **163**

Introduction

> "Highways, telephone lines, electric power systems, computer chips, water delivery systems, and rail lines: these physical networks, and many others, are familiar to all of us. In each of these problem settings, we often wish to send some good(s) (vehicles, messages, electricity, or water) from one point to another, typically as efficiently as possible—that is, along a *shortest route* or via some *minimum cost flow pattern*."
>
> Ahuja, Magnanti, Orlin & Reddy [1995]

The shortest path problem is one of the most commonly encountered problems in the study of transportation and communication networks. A quick look at the paper "Applications of Network Optimization" by Ahuja, Magnanti, Orlin & Reddy [1995] should suffice to convince oneself of the enormous range of applications of shortest path methods. The shortest path problem is also one of the most elementary problems and possibly one of the most fundamental problems in network optimization. Many combinatorial optimization and network algorithms uses shortest paths algorithms as important subroutines (see, e.g. Ahuja, Magnanti & Orlin [1993]).

The theme of this thesis is the application of shortest path methods to a number of practical problems, with special attention to the analysis of the complexity of the problems and of the efficiency of the methods.

The first chapter of this thesis deals with the aircraft assignment problem. This problem consists of assigning aircrafts of an airline company to flight legs in order to carry out a pre-specified weekly timetable. To solve the problem, Schrijver [1993] translate it to the problem of finding disjoint paths in an acyclic directed graph. This latter problem is modeled as a shortest path problem via a well-known reduction due to Perl & Shiloach [1978]—later extended by Fortune, Hopcroft & Wyllie [1980].

VLSI-routing can be mathematically formulated as the problem of finding disjoint paths in a planar graph. Finding disjoint paths in planar graphs is in general a difficult problem (i.e. $\mathcal{NP}$-hard; cf. Lynch [1975]). In some special cases, however, efficient algorithms for the problem are known. Schrijver [1991] presented an efficient algorithm (i.e. polynomially bounded time) that finds vertex-disjoint paths in a planar graph according to a prescribed "sketch" (a sketch in the VLSI framework is a specification of the topological placement of the wires interconnecting the nets with respect to the modules). In Chapter 2 we describe Schrijver's algorithm. Shortest paths algorithms arise as an important subroutine within this algorithm.

In one of the steps of Schrijver's disjoint paths algorithm described in Chapter 2 a system $Ax \leq b$ must be solved in integers, where b is an integral m-vector and A is an $m \times n - \{0, \pm 1\}$-matrix with at most two nonzero entries per row. Using Fourier-Motzkin elimination of variables Schrijver showed that such a system can be solved in integers in polynomial time. In Chapter 3 we reduce the solution of $Ax \leq b$ in integers to a shortest path computation.

In Chapter 4 we consider the problem $\max\{c^T x \mid Ax \leq b,\ x \text{ integral}\}$, where A is a $\{0, \pm 1\}$-matrix as in Chapter 3. We show that this problem can be solved by a number of shortest paths computations provided that $c \geq 0$ and matrix A has no more than one positive element in any row (the deletion of any of these constraints makes the problem $\mathcal{NP}$-hard).

One issue in reliability of communication networks is finding many disjoint routes between an origin station and a destination station. There are at least two possible optimization problems for this type of routing problem: (*i*) the *min-sum problem* consisting of minimizing the sum of the delays of the routes (minimum cost routing); and (*ii*) the *min-max problem* consisting of minimizing the time at which the last transmission is completed. The problem of finding disjoint min-sum and min-max routes in a communication network can be modeled as the problem of finding "shortest" and "length-bounded" disjoint paths in an undirected graph. These problems turned out to be much harder than just finding disjoint paths; for instance, even simple cases where finding disjoint paths is easy are $\mathcal{NP}$-hard for finding length-bounded disjoint paths. In Chapter 5 we use dynamic programming techniques in order to show the polynomial or pseudo-polynomial time solvability of the shortest and length-bounded disjoint paths problem in planar graphs with certain special structures. These dynamic programming techniques can be seen as the problem of finding a shortest path in a (large) acyclic directed graph.

The contents of Chapter 6 constitutes a motivation for the topic treated in Chapter 7. In this chapter we discuss the train timetabling problem and we describe a backtracking approach to find a periodic train timetable that fulfills a given set of constraints. This backtracking approach can be sped up if a shortest circuit basis of a certain graph is provided. (A basis of the circuit space of a graph is said a *shortest circuit basis* if the sum of its circuits length is

minimum.) The backtracking approach described is due to Schrijver [1994] and the train timetable model we consider is due to Schrijver & Steenbeek [1994].

In Chapter 7 we focus on the shortest circuit basis problem. We describe two polynomial-time algorithms for finding a shortest circuit basis of a graph (one of them due to Horton [1987]). Both algorithms call upon shortest path algorithms as fundamental subroutines. In this chapter we also discuss the close relation between bases of circuits and odd circuits in signed graph and we study the connection between bases of circuits and polyhedral theory. An application of a shortest circuit basis in analysis of electrical networks is given.

In Chapter 8 we study the shortest cut basis problem. This problem is the dual (in the circuit-cut sense) of the shortest circuit basis problem. In this chapter we provide dual results to the ones obtained in Chapter 7 (thus the relation between shortest paths and the subject of this chapter is via duality).

In our final chapter we generalize the results of Chapter 7 and 8 to binary matroids.

We mention at this point that the results presented in this thesis were obtained in collaboration with A.M.H. Gerards (Chapters 8 and 9), H. van der Holst (Chapter 5) and A. Schrijver (Chapters 3, 5, 7, 8 and 9).

Preliminaries

We summarize here the background material and notation we shall use in this thesis. We start with some general preliminaries. In Section 2 we present the basic terminology needed in graph theory. Next, in Section 3, we briefly review computational complexity theory. Finally, in Section 4 we discuss the shortest path problem.

1. General preliminaries

We denote the sets of reals, rationals, and integers by $\mathbb{R}$, $\mathbb{Q}$ and $\mathbb{Z}$, respectively. $\mathbb{R}_+$, $\mathbb{Q}_+$ and $\mathbb{Z}_+$ are the sets of nonnegative reals, rationals and integers, respectively.

Let E be a finite set. We consider each $x \in \mathbb{R}^E$ as a vector indexed by E, i.e. $x = (x_e)_{e \in E}$. For each subset $B \subseteq E$, we define its *incidence vector* $\chi^B = (\chi^B_e)_{e \in E}$ by

$$\chi^B_e := \begin{cases} 1 & \text{if } e \in B; \\ 0 & \text{otherwise.} \end{cases}$$

For $x \in \mathbb{R}^E$ its l_∞-norm is $||x||_\infty := \max_{e \in E} |x_e|$.

The *convex hull* of a set X of vectors, denoted by conv.hull X, is the set

$$\begin{aligned} \text{conv.hull } X \quad := \quad & \{\lambda_1 x_1 + \cdots + \lambda_t x_t \mid t \geq 1;\ x_1, \ldots, x_t \in X; \\ & \lambda_1, \ldots, \lambda_t \geq 0;\ \lambda_1 + \cdots + \lambda_t = 1\}. \end{aligned}$$

The *cone generated* by a set X of vectors is defined as

$$\begin{aligned} \text{cone } X \quad := \quad & \{\lambda_1 x_1 + \cdots + \lambda_t x_t \mid t \geq 0;\ x_1, \ldots, x_t \in X; \\ & \lambda_1, \ldots, \lambda_t \geq 0\}. \end{aligned}$$

A set P of vectors in $\mathbb{R}^E$ is a polyhedron if $P = \{x \in \mathbb{R}^E \mid Ax \leq b\}$, where A and b are a matrix and a vector of appropriated dimension. A vector x is

said a *vertex* of a polyhedron P if $x \in P$ and there do not exist distinct vectors $y, z \in P$ and reals $\lambda, \mu > 0$, $\lambda + \mu = 1$ so that $x = \lambda y + \mu z$.

For sets X and Y, $X \Delta Y$ denotes $(X \setminus Y) \cup (Y \setminus X)$, i.e. the *symmetric difference* of X and Y.

2. Graph theory

We present here the basic terminology in graph theory we shall need along this thesis. Most of this terminology will follow Bollobás [1979], Bondy & Murty [1976] and Wilson [1976].

Undirected graphs

An *undirected graph* (or simply graph) G is a pair $(V(G), E(G))$, where $V(G)$ is a finite set of elements called *vertices* and $E(G)$ is a finite multi-set of unordered pairs of $V(G)$ called *edges*. An edge $e = \{u, v\}$ (also denoted by $e = uv$) is said to *connect* the vertices u and v; the vertices u and v are called the *endpoints* of e. An edge with identical endpoints is called a *loop* and an edge e is said *parallel* to an edge e' ($e \neq e'$) if e and e' connect the same pair of different vertices. We shall say that two vertices u and v of G are *adjacent* if G has an edge e connecting u and v; the edge e is said to be *incident* with u and with v.

A graph is usually represented pictorially in the plane. Each vertex is indicated by a point and each edge by a line joining the points the represent the endpoints of that edge. A *planar graph* is a graph which can be *embeddable* in the plane, that is, it can be drawn in the plane so that its edges intersect only at their endpoints.

A graph H is a *subgraph* of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. The subgraph of G *induced* by a vertex set $V' \subseteq V(G)$ is the subgraph of G whose vertex set is V' and the edge set is the set of edges of G with both endpoints in V'; we denote this subgraph by $G[V']$. The subgraph $G[V \setminus V']$ is also denoted by $G - V'$. Now, the subgraph of G *induced* by an edge set $E' \subseteq E(G)$, denoted by $G[E']$, is the subgraph of G whose edge set is E' and the vertex set is the set of endpoints of edges in E'. The subgraph $G[E \setminus E']$ (also denoted by $G - E'$) is said the subgraph resulting from the *deletion* of E'.

A *path* in a graph G is a non-null sequence

$$(v_0, e_1, v_1, e_2, \ldots, v_{t-1}, e_t, v_t), \tag{0.1}$$

where $v_0, v_1, \ldots, v_t$ are vertices of G and $e_1, \ldots, e_t$ are edges of G so that $e_i = \{v_{i-1}, v_i\}$ ($i = 1, \ldots, t$). The vertices v_0 and v_t are the *endpoints* of the path (0.1). We call path (0.1) a $v_0 - v_t$-path and say that it *connects* v_0 and v_t. The vertices $v_1, \ldots, v_{t-1}$ are said the *internal vertices* of the path (0.1). Path (0.1) is *simple* if all vertices and edges in (0.1) are distinct.

If $v_0 = v_t$ then (0.1) is called a *cycle*. In addition, if (0.1) has no repeated

edges or vertices, besides v_0 and v_t, then (0.1) is called a *circuit*. A *Hamiltonian circuit* is a circuit containing each vertex of G.

Two vertices u and v of G are said to be *connected* if G contains a $u-v$-path. Connectedness induces an equivalence relation on the vertex set of G. The classes of this equivalence relation are called the *connected components* (or simply components) of G. If any pair of vertices of G are connected then G is said a *connected graph*. A vertex v is called a *cut vertex* if the number of components of $G - \{v\}$ is larger than the number of components of G. A connected graph with no cut vertex is said a *block* or *2-connected*.

A graph without any cycle is a *forest*; a *tree* is a connected forest. A subgraph T of G is called a *spanning tree* if T is a tree and $V(T) = V(G)$. If T^* is a subgraph of G so that $G - E(T^*)$ is connected then T^* is said a *cotree*. A maximal (with respect to the edge set) cotree is said to be a *spanning cotree*.

For nonempty proper subsets X and Y of the vertex set of G we denote by $\delta_G(X,Y)$ the set of edges with one endpoint in $X \setminus Y$ and the other endpoint in $Y \setminus X$. We use $\delta_G(X)$ for $\delta_G(X, V \setminus X)$; such a set of edges is called a *cocycle*. A minimal nonempty cocycle of G is called a *cut*. Let $d_G(X,Y) := |\delta_G(X,Y)|$ and $d_G(X) := |\delta_G(X)|$.

We denote by G/e the graph resulting by the *contraction* of the edge e of G, that is the graph resulting from $G - \{e\}$ by identifying the endpoints of e. If $E' = \{e_1, \ldots, e_t\} \subseteq E$ then G/E' is an abbreviation of $(((G/e_1)/e_2)\ldots)/e_t)$ (here we are using the fact that "contraction of an edge" is a commutative operation). For $V' \subseteq V$, we denote by G/V' the graph $G/E(G[V'])$. G' is said a *minor* of G if it can be obtaining from G by a series of deletions and contractions.

Let $P = (v_0, e_1, v_1, \ldots, e_t, v_t)$ and $Q = (w_0, f_1, w_1, \ldots, f_k, w_k)$ be paths of G so that $v_t = w_0$. The *concatenation* of the paths P and Q, denoted by $P \cdot Q$, is the path $(v_0, e_1, v_1, \ldots, e_t, v_t = w_0, f_1, w_1, \ldots, f_k, w_k)$

Vector spaces associated with undirected graphs

The *circuit space* of a graph $G = (V,E)$ is the linear subspace of $\mathrm{GF}(2)^E$ generated by the characteristic vectors of the circuits of G. The *cut space* of G is defined similarly.

We say that the set of circuits $\{C_1, \ldots, C_\nu\}$ forms a *basis of circuits* of G if the incidence vectors of these circuits form a basis of the circuit space of G. Analogously we define *basis of cuts*.

Let G be a 2-connected graph, T and T^* be a spanning tree and a spanning cotree of G, respectively. For each edge $e \in E(G) \setminus E(T)$ the subgraph $T + e$ of G contains a unique circuit $C(T,e)$ of G called the *fundamental circuit of* e (with respect to T). For each edge $e \in E(G) \setminus E(T^*)$ the subgraph $T^* + e$

contains a unique cut $C^*(T^*, e)$ of G called the *fundamental cut of* e (with respect to T^*). It was first observed by Kirchhoff [1847] that $\{C(T,e) \mid e \in E(G)\backslash E(T)\}$ forms a circuit basis of G and that $\{C^*(T^*, e) \mid e \in E(G)\backslash E(T^*)\}$ forms a cut basis of G. It is well-known that circuit space and the cut space are orthogonal complement of each other.

Directed graphs

A *directed graph* or *digraph* D is a pair $(V(D), A(D))$, where $V(D)$ is a finite set of vertices and $A(D)$ is a finite multi-set of ordered pairs of $V(D)$ called *arcs*. If $a = (u, v)$ is an arc of D then u is said the *tail* of a and v is said the *head*. Pictorially the arcs of a digraph are represented by arrows.

For a nonempty proper subset S of the vertex set of D we denote by $\delta_D^+(S)$ the set of arcs entering S (i.e. the set of arcs whose head is in S) and by $\delta_D^-(S)$ the set of arcs leaving S.

Corresponding to a digraph D there is a graph G on the same vertex set having for each arc of D an edge with the same endpoints. This graph is the *underlying graph* of D.

A (*directed*) *path* is a non-null sequence

$$(v_0, a_1, v_1, a_2, \ldots, v_{t-1}, a_t, v_t), \tag{0.2}$$

where $v_0, v_1, \ldots, v_t$ are vertices of D and $a_1, \ldots, a_t$ are arcs of G so that $a_i = (v_{i-1}, v_i)$ $(i = 1, \ldots, t)$. Path (0.2) is said a path from v_0 to v_t or a $v_0 - v_t$-path. If $v_0 = v_t$ then (0.2) is called a *directed cycle* and if in addition (0.2) has no repeated vertices or arcs, besides v_0 and v_t, then (0.2) is a *directed circuit*.

We associate, in a natural way, with each circuit of the underlying graph of D a (undirected) circuit of D containing forward and backward arcs.

A digraph with no directed cycle is said *acyclic*. A *topological ordering* of the vertices of D is a bijective function $f : V(D) \to \{1, 2, \ldots, |V(D)|\}$ so that $(u, v) \in A(D)$ implies $f(u) < f(v)$. A digraph admits a topological ordering if and only if it is acyclic. An topological ordering of an acyclic graph can be found by breadth-first search (see, e.g., Ahuja, Magnanti & Orlin [1993: Ch. 1]).

A vertex v is said *reachable* from a vertex u if D has a $u - v$-path.

Two vertices u and v are said *connected* if D has a $u - v$-path and a $v - u$-path. Again, connectedness induces an equivalence relation on the vertex set of D whose classes are the *strongly-connected components* (or simply *strong components*) of D. D is said *strongly connected* if any two vertices of D are reachable from each other.

The *incidence matrix* of a digraph $D = (V(D), A(D))$ is the $V(D) \times A(V)$-matrix N so that, for each $v \in V(D)$ and $a \in A(D)$, $N_{v,a} = 1, -1, 0$

depending on whether a is the head, tail or is non-incident to v, respectively.

Vector spaces associated with digraphs

To each circuit C of a digraph D we associate an arbitrary orientation and a partition (C^+, C^-) of its arc set so that the arcs of C in C^+ (C^-, respectively) are the arcs whose direction agrees (disagrees, respectively) with the orientation of C. Now, we associate with each circuit C the following vector $z_C \in \{0, \pm 1\}^{A(D)}$:

$$z_C(a) := \begin{cases} 1 & \text{if } a \in C^+; \\ -1 & \text{if } a \in C^-; \\ 0 & \text{otherwise.} \end{cases}$$

The *circuit space* of a digraph D is the linear subspace of $\mathbb{R}^{A(D)}$ spanned by the vectors z_C, C a circuit of D.

Similarly, if S is a nonempty proper subset of the vertex set of a digraph D then we associate with S a vector $u_S \in \{0, \pm 1\}^{A(D)}$ so that:

$$u_S(a) := \begin{cases} 1 & \text{if } a \in \delta_D^+(S) \\ -1 & \text{if } a \in \delta_D^-(S) \\ 0 & \text{otherwise.} \end{cases}$$

The *cut space* of a digraph D is the linear subspace of $\mathbb{R}^{A(D)}$ spanned be the vectors z_S, $\emptyset \neq S \subset V(D)$.

We say that the set of (undirected) circuits $\{C_1, \ldots, C_\nu\}$ is a *basis of circuits* of D if $\{z_{C_1}, \ldots, z_{C_\nu}\}$ forms a basis of the circuit space of D. A *basis of cuts* of a digraph D is defined analogously.

Analogously to the "undirected case", $\mathbb{R}^{A(D)}$ can be seen to be directed sum of the circuit space and cut spaces of D.

Signed graphs

A *signed graph* is a pair (G, Σ), where $G = (V, E)$ is an undirected graph and $\Sigma \subseteq E$. We call Σ a *sign* (of the edges) of G. The edges in Σ are called *odd* edges and the remaining edges are called *even* edges. A cycle of G is said an *odd cycle* (*even cycle*, respectively) in (G, Σ) if it contains an odd number (even number, respectively) of odd edges.

3. Computational complexity

For a thorough introduction to the subject treated (rather informally) in this section see, for instance, the books Aho, Hopcroft & Ullman [1974], Garey & Johnson [1979], Knuth [1968], Papadimitriou [1994] and Papadimitriou & Steiglitz [1982] and the papers Lenstra & Rinnooy Kan [1979] and Tarjan [1978].

In this thesis we study the solution of some *combinatorial optimization problems* by *efficient algorithms*. In this section is our intention to set up

the necessary background and nomenclature in computational complexity and algorithm analysis.

Problems and algorithms

Informally, a *problem* is a question to be solved or decided. Examples of problems are:

$$\begin{array}{rl} \textit{given}: & \text{– an undirected graph } G=(V,E); \\ \textit{question}: & \text{– does } G \text{ contain a Hamiltonian circuit?} \end{array} \tag{0.3}$$

$$\begin{array}{rl} \textit{given}: & \text{– an undirected graph } G=(V,E); \text{ and} \\ & \text{– vertices } s \text{ and } t \text{ of } G; \\ \textit{find}: & \text{– a path of } G \text{ connecting } s \text{ to } t, \text{ if there is any.} \end{array} \tag{0.4}$$

$$\begin{array}{rl} \textit{given}: & \text{– an undirected graph } G=(V,E); \text{ and} \\ & \text{– a "weight" function } w: E \to \mathbb{Z}; \\ \textit{find}: & \text{– a spanning tree } T \text{ of } G \text{ so that} \\ & \displaystyle\sum_{e \in E(T)} w(e) \text{ is as small as possible.} \end{array} \tag{0.5}$$

Problems (0.3), (0.4) and (0.5) are known as the *Hamiltonian circuit problem*, the *reachability problem* and the *minimum spanning tree* (MST) *problem*, respectively.

The set of all possible inputs to a problem is its *instance set*. Thus, any particular graph $G=(V,E)$ is an instance of the Hamiltonian circuit problem and any particular graph $G=(V,E)$ and function $w: E \to \mathbb{Z}$ is an instance of the minimum spanning tree problem.

A problem which asks for a *yes/no* answer is called a *decision problem*. A problem searching for an element in a set S is said a *search problem*. Finally, a problem seeking for an element in a set S which is best possible with respect to some criterion is a *optimization problem*. Thus, the Hamiltonian circuit, reachability and the minimum spanning tree problem are examples of a decision, a search and an optimization problem, respectively. Whenever the set S is finite the respective problem is said to be a *combinatorial problem*.

An *algorithm* is a *recipe*, *method*, *process*, a finite sequence of *instructions* or *operations* to solve a problem. (See Knuth [1968: Ch. 1] for a discussion on the features an algorithm must satisfy.)

Machine models

In this thesis we use as model of computation a *random access machine* (cf. Cook & Reckhow [1973]—see also Aho, Hopcroft & Ullman [1974: Ch. 1]). A

random access machine (RAM) is an abstraction of a general purpose digital computer. Typical *elementary operations* of a RAM are: addition, subtraction, branching, comparison, multiplication and division.

The random access machine provides a realistic tool for measuring the complexity of an algorithm, with one serious drawback; namely, its memory's cells and registers can hold an arbitrary large integer (so, such a machine can, in principle, add arbitrary large integers in one elementary operation—which is a quite unrealistic assumption). We can avoid this problem by charging for each operation a time proportional to the number of bits needed to represent the operands (*logarithmic cost measure—cf.* Cobham [1965] and Aho, Hopcroft & Ullman [1974: Ch. 1]). Alternatively, we can limit the capacity of each memory's cell and register to hold integers whose *size* (i.e. the number of bits necessary to represent it) is bounded by a polynomial in the *problem size* (i.e. the number of bits of the input). We shall use the latter approach and count each elementary operation as one step (*uniform cost measure*).

Complexity measure and polynomial-time algorithms

Once we have chosen a computer model we may now proceed and select our complexity measure. In analyzing the complexity of an algorithm (or of its expression as a RAM program) we will mainly be interested in measuring its *running time* (i.e., the number of elementary operations) as a function of the problem size. In order to fix the ideas consider the MST problem. The size of the MST problem under a "reasonable" encoding scheme (cf. Garey & Johnson [1979: Ch. 1]) is proportional to $\sigma(G,w) := |V| + |E| + \sum_{e \in E} \lceil \log(|w(e)| + 1) \rceil$. Hence, we express the running time of any algorithm that solves the MST problem as a function of $\sigma(G,w)$. The *storage space* of an algorithm will be measured in the same fashion as its running time.

In our expression of the running time of an algorithm we will ignore constant factors and we shall consider the algorithm's asymptotic behavior (its behavior for problems of "large enough" sizes). More precisely, let P be a problem, $\mathcal{I}$ be its instance set and $\sigma : \mathcal{I} \to \mathbb{Z}_+$ so that $\sigma(I)$ is the size of I for all $I \in \mathcal{I}$. We say that an algorithm A solves P in $O(\phi_{\mathsf{A}}(\sigma(I)))$ time for an real-valued function ϕ_{A} if there exists a constant c such that the running time of A is bounded above by $c\phi_{\mathsf{A}}(\sigma(I))$ for each $I \in \mathcal{I}$. In this case we also say that A *runs* in $O(\phi_{\mathsf{A}}(\sigma(I)))$ time. An algorithm A is said a *polynomial-time* algorithm whenever for all $I \in \mathcal{I}$ we have that $\phi_{\mathsf{A}}(\sigma(I)) \leq p(\sigma(I))$ for a polynomial function p. This corresponds to the so-called *worst-case analysis* of an algorithm, since we are bounding the algorithm's running time as a function of the worst-case input data.

A *linear-time* algorithm is a algorithm whose running time is proportional to the problem size.

A criterion of efficiency ($\mathcal{P}$ and $\mathcal{NP}$)

By an *efficient* or *good* algorithm we mean a polynomial-time algorithm. This criterion of efficiency was independently introduced by Cobham [1965] and Edmonds [1965]. A problem is said to be *tractable* if there exists an efficient algorithm that solves it and is said *intractable* otherwise. The class of all tractable decision problems is denoted by $\mathcal{P}$. Examples of problems in $\mathcal{P}$ are the decision versions of: (i) the reachability problem (i.e., *given*: an undirected graph $G = (V, E)$ and two vertices s and t of G; *question:* does G has a path connecting s and t?); and (ii) the MST problem (i.e., *given*: an undirected graph $G = (V, E)$; $w : E \to \mathbb{Z}$ and $L \in \mathbb{Z}$; *question*: does G have a spanning tree T such that $\sum_{e \in T} w(e) \leq L$?). The reachability problem can be solved in $O(|E|)$ time by breadth-first search (see, e.g., Tarjan [1983]) and the MST problem can be solved in $O(m\beta(m, n))$ time by an algorithm due to Fredman & Tarjan [1987], where $\beta(m, n) := \min\{i \mid log^{(i)} n \leq m/n\}$.

A class of decision problems possibly bigger than $\mathcal{P}$ is $\mathcal{NP}$. A decision problem is in $\mathcal{NP}$ whenever there exists a short proof (i.e. a polynomial-length proof) of the answer "yes". For instance, if the Hamiltonian circuit problem has answer yes for the instance $G = (V, E)$ then one can prove it by exhibiting a Hamiltonian circuit of G, the size of such a circuit is proportional to $|V|$ (thus, the size of the circuit is $O(|V|)$), a polynomial in the problem size. Hence, the Hamiltonian circuit problem belongs to $\mathcal{NP}$. From the definition of $\mathcal{NP}$ it is clear that $\mathcal{P} \subseteq \mathcal{NP}$, since the sequence of computations of a polynomial-time algorithm that solves a problem in $\mathcal{P}$ can be taken as a short proof of the answer (whatever that is).

Cook [1971] showed the existence within $\mathcal{NP}$ of a class of problems which are at least as hard as any problem in $\mathcal{NP}$, the so-called $\mathcal{NP}$-*complete* problems. More specifically, a decision problem P is $\mathcal{NP}$-complete if it satisfies two properties:

(i) P is in $\mathcal{NP}$; and

(ii) if Q is in $\mathcal{NP}$ then by also considering a call to a subroutine which solves problem P as an elementary operation one can solve Q in polynomial time—this corresponds to the concept of polynomial time Turing reducibility or Cook-reducibility (cf. Garey & Johnson [1979: Ch. 5])).

Hence, if a $\mathcal{NP}$-complete problem is polynomial solvable then $\mathcal{P} = \mathcal{NP}$. The Hamiltonian circuit problem is an example of a $\mathcal{NP}$-complete problem (cf. Karp [1972]). Any problem (not necessarily a decision problem) that is at least as hard as any problem in $\mathcal{NP}$ is said a $\mathcal{NP}$-*hard* problem.

For a simple "tale-based" introduction to $\mathcal{P}$ and $\mathcal{NP}$—but with the whole essence—see Lovász & Plummer [1986: Ch. 1].

Strongly-polynomial time algorithms, pseudo-polynomial time algorithms and $\mathcal{NP}$-completeness in the strong sense

A polynomial-time algorithm is said *strongly polynomial* if its running time does *not* depend on the size of the numbers occurring in the input (if there is any). The algorithm due to Fredman & Tarjan [1987] for the MST problem is an example of a strongly polynomial algorithm, since its running time ($O(m\beta(m,n))$) does not depend on the weight on the edges. Examples of polynomial-time algorithms that are *not* strongly polynomial are: (i) the Euclidean algorithm for finding the greatest common divisor of two positive integers α and β whose running time is $O(\log\alpha + \log\beta)$ (see Schrijver [1986: Ch. 5]); and (ii) Khachiyan's algorithm to solve a system of linear inequalities $Ax \leq b$ in $O(n^5 \log L)$ time, where A is an integral $m \times n$-matrix, $b \in \mathbb{Z}^m$ and L is the maximum absolute value of the entries in A, b (cf. Khachiyan [1979]—see also Grötschel, Lovász & Schrijver [1988: Ch. 3], Papadimitriou & Steiglitz [1982: Ch. 8], Schrijver [1986: Ch. 13]).

Let P be a problem and $\mathcal{I}$ be its instance set. For each $I \in \mathcal{I}$ we denote by $\sigma(I)$ and $\mathrm{Max}(I)$ the size of I and the magnitude of the largest integer appearing in I, respectively. We define $\mathrm{Max}(I) := 0$ if no integer occurs in I. A *pseudo-polynomial* time algorithm for P is an algorithm whose running time is bounded above by a polynomial function of the two variables $\sigma(I)$ and $\mathrm{Max}(I)$.

Let $\mathcal{I}_f := \{I \in \mathcal{I} \mid \mathrm{Max}(I) \leq f(\sigma(I))\}$ for an integer-valued function f on $\mathcal{I}$ and let P_f be the problem P restricted to the instance set $\mathcal{I}_f$. P is *$\mathcal{NP}$-complete in the strong sense* or *strongly $\mathcal{NP}$-complete* if P_p is $\mathcal{NP}$-complete for some polynomial p. Consider the following *symmetric traveling salesman problem* (TSP):

given : – an undirected graph $G = (V, E)$;
– a "cost" function $c : E \to \mathbb{Z}_+$;
– an nonnegative integer B;

question : – does G have a Hamiltonian circuit C such that (0.6)

$$\sum_{e \in C} c(e) \leq B?$$

The TSP can be easily seen to be $\mathcal{NP}$-complete in the strong sense, since the TSP restricted to instances such that $c(e) = 1$ and $B = |V| - 1$ corresponds to the Hamiltonian circuit problem. For more on pseudo-polynomial algorithms and $\mathcal{NP}$-completeness in the strong sense see Garey & Johnson [1979: Ch. 4] and Papadimitriou & Steiglitz [1982: Ch. 16].

4. The shortest path problem

In this section we summarize some well-known results on the shortest path problem. For a more extensive treatment we refer to Ahuja, Magnanti & Or-

lin [1993: Ch. 4 and 5], Lawler [1976: Ch. 3] and Tarjan [1983: Ch. 7]. For a survey see Dreyfus [1969] and Deo & Pang [1984].

Let $D = (V, A)$ be a directed graph whose arcs have integer-valued (possibly negative) *lengths*. We denote the length of an arc (u, v) of D by $\text{length}_D(u, v)$. A *shortest path* from a vertex u to a vertex v is a directed path P from u to v whose sum of the arcs length is minimum. In this thesis we are interested in two particular types of shortest path problems, namely:

The single-source shortest path problem: *given*: a digraph $D = (V, A)$ with length on its arcs and a source $s \in V$; *find:* for any vertex v a shortest path, i.e. a path of minimum length, from s to v.

The all-pairs shortest path problem: *given*: a digraph $D = (V, A)$ with length on its arcs; *find:* for every pair of vertices s and t a shortest path from s to t.

The following propositions give well-known properties of shortest paths. A *negative-length cycle* is a cycle the sum of whose arc lengths is negative.

Proposition 0.1 *Let s and t be vertices of a digraph D so that t is reachable from s. Then*
either *D has a shortest path from s to t;*
or *D has a negative-length cycle C such that C is reachable from s and t is reachable from C.*
Moreover, if D has a shortest $s - t$-path then it has a simple one. ∎

A *shortest-path tree* rooted at a vertex s of D is a spanning arborescence rooted at s each of whose paths is a shortest path, in D.

Proposition 0.2 *Let s be a vertex of a digraph D so that each vertex of D is reachable from s. Then*
either *D has a shortest-path tree rooted at s;*
or *D has a negative-length cycle.*
∎

The distance from a vertex u to a vertex v in D, denoted by $\text{distance}_D(u, v)$, is the length of a shortest $u - v$-path in D.

Proposition 0.3 *T is a shortest-path tree with root s if and only if, for every arc (u, v) of D, $\text{distance}_D(s, u) + \text{distance}_D(u, v) \geq \text{distance}_T(s, v)$.* ∎

We shall denote by n and m the number of vertices and the number of arcs of D, respectively.

The single-source shortest path problem

Let D be a digraph with nonnegative arc lengths. The computationally most efficient algorithm for the single-source shortest path problem was designed first by Dijkstra [1959] and independently by Whiting & Hillier [1960]. The running time of Dijkstra's original implementation is $O(n^2)$. Making use of a data structure called "heap" (or "priority queue", see, e.g., Tarjan [1983: Ch. 3]) Johnson [1977] proposed a faster implementation of Dijkstra's algorithm for sparse graphs. Johnson's implementation runs in $O(m \log n)$ time. Fredman & Tarjan [1987] have developed a variant of the heap data structure (the so-called "Fibonacci heap") which can be seen to implement Dijkstra's algorithm in time $O(m + n \log n)$. Fredman & Tarjan's implementation of Dijkstra's algorithm achieves the best strongly polynomial-time bound for solving the single-source shortest path problem on digraphs with nonnegative arc lengths.

Dial [1969] (and independently Wagner [1976]) proposed a pseudo-polynomial time implementation of Dijkstra's algorithm which runs in $O(m + nC)$ time, where C is the maximum length of an arc.

Summarizing the running time of different implementations of Dijkstra's algorithm we have:

Running time	*Reference*
$O(n^2)$	Dijkstra [1959]
$O(m \log n)$	Johnson [1977]
$O(m + n \log n)$	Fredman & Tarjan [1987]
$O(m + nC)$	Dial [1969], Wagner [1976]
$O(m + n\sqrt{\log C})$	Ahuja, Mehlhorn, Orlin & Tarjan [1990]

(0.7)

The single-source shortest path problem on digraphs with arbitrary arc lengths is a more difficult problem. After more than three decades Bellman-Ford-Moore is still the best strongly polynomial-time algorithm for solving this problem (cf. Bellman [1958], Ford [1956], Moore [1959]). Bellman-Ford-Moore's algorithm runs in $O(mn)$ time.

Polynomial time algorithms using scaling techniques (an approach introduced by Edmonds & Karp [1972]) were developed by Gabow [1985] and by Gabow & Tarjan [1989]. The former algorithm runs in time $O(n^{3/4} m \log C)$; the latter algorithm runs in time $O(\sqrt{n} m \log(nC))$. Here, C is the maximum absolute value of the length of an arc. Goldberg [1993], also applying scaling techniques, designed an algorithm which runs in time $O(\sqrt{n} m \log L)$, where L is a positive integer (greater than 1) so that the arcs length are at least $-L$. As $L \leq C$ the bound of Goldberg's algorithm dominates the one of Gabow & Tarjan. Thus, the current best running time for solving the single-source shortest path problem on digraphs with arbitrary arc lengths is $O(\min\{nm, \sqrt{n} m \log L\})$.

Finally, in an acyclic digraph with arbitrary arcs length the single-source

shortest path problem can be solved in $O(m)$ time by "scanning" the vertices in topological ordering.

The all-pairs shortest path problem

Floyd [1962] obtained an algorithm to solve the all-pairs shortest path algorithm in a digraph D with arbitrary arc lengths by generalizing the transitive closure algorithm of Warshall [1960]. Floyd-Warshall's algorithm runs in $O(n^3)$ time. By combining the algorithms for the single-source shortest path problem of Bellman-Ford-Moore and Fredman & Tarjan [1987] one can obtain an algorithm which for sparse graph is faster than the Floyd-Warshall method. Indeed, by one application of Bellman's algorithm one can compute in time $O(nm)$ a shortest path tree rooted at an arbitrary vertex s of D (we may assume all vertices of D are reachable from s). As it is well-known, shortest paths in D are invariant under the following length transformation

$$\text{length}'_D(u,v) := \text{distance}_D(s,u) + \text{length}_D(u,v) - \text{distance}_D(s,v),$$

for each (u,v) arc of D. The transformed length function is nonnegative. Now, with n applications of Fredman & Tarjan's shortest path algorithm one can solve the all-pairs shortest path problem for the graph with transformed arc lengths (given a solution to the original problem). The overall running time of this algorithm is $O(n(m + n\log n))$.

Shortest paths in planar graphs

In planar graphs all algorithms above have automatically better bounds (as $m = O(n)$). However, one can do better than simply use the sparsity of planar graphs. Making use of a graph-decomposition based on a "separators" Frederickson [1987] designed an algorithm for the single-source problem in planar graphs with nonnegative arc lengths which runs in time $O(n\sqrt{\log n})$ (beating Johnson's algorithm by a factor of $\sqrt{\log n}$). In his paper Frederickson also proposed an $O(n^2)$-time algorithm for the all-pair shortest path problem in planar graphs with (possibly) negative arc lengths but *no* negative cycles. More recently, Klein, Rao, Rauch & Subramanian [1994] proposed a linear-time algorithm for the single-source shortest path problem in planar graphs with nonnegative arc lengths.

Notation

For convenience we denote the functions expressing the time complexity for solving the single-source shortest path problem and the all-pairs shortest path problem in a graph with m arcs and n vertices by $T_{ss}(m,n)$ and $T_{all}(m,n)$, respectively. Thus, both $T_{ss}(m,n)$ and $T_{all}(m,n)$ depend on the particular features of the graph we shall be considering.

Chapter 1

Aircraft assignment and disjoint paths in acyclic digraphs

1.1. Introduction

An airline company has a pre-specified weekly timetable to be carried out by a fleet of k aircrafts $p_1, \ldots, p_k$ of possibly different types (e.g. Boeing 747s or McDonnell Douglas DC 10s or MD 11s). Every week the airline has to specify the flight legs that should be performed by each aircraft during the forthcoming week (by forthcoming week we mean from Sunday 0:00h till Saturday 24:00h). Each entry of the weekly timetable consists of a flight number i (e.g. 0837), a departure city dc_i (e.g. Amsterdam), a departure time dt_i (e.g. Sunday 20:15h), an arrival city ac_i (e.g. Singapore) and an arrival time at_i (e.g. Monday 14:50h).

The timetable is such that at any moment during the next week the number of aircrafts on the ground at any particular city is sufficient to perform an outgoing flight from that city. Moreover, the number of incoming flights during the whole week at any specific city equals the number of outgoing flights.

This implies that the task of assigning the aircrafts to the flight legs would be trivial if it were not complicated by complex rules. (Indeed, one could greedily pick up any grounded aircraft on city dc_i at the moment dt_i to perform flight i.) An aircraft is clearly not available for a flight immediately after its arrival; some time is needed for disembarking and routine maintenance. This constraint, in particular, can be overcome by adding the require disembarking/maintenance time to the arrival time of each flight (so, the aircraft that carries out the flight i is ready for carrying out the next flight leaving city ac_i from at_i on).

The rule that gives complications is a legal one. After a certain number of flight hours an aircraft must undergo a "big" maintenance which is made only at the airline home base. This maintenance takes a day (9.00h-18.00h) and is

made regularly from Monday till Friday. Thus, next week some aircrafts must be on the ground at the home city for maintenance during one of these days from 9:00h till 18:00h. In the sequel we shall consider "undergo maintenance at home base" as a flight; more specifically, if the maintenance of aircraft must be made on (say) Tuesday then we consider this as a flight having as departure city the home city, as departure time Tuesday 9:00h, as arrival city the home city and as arrival time Tuesday 18:00h.

One more possible constraint is that for certain flights an aircraft with more seats or cargo area may be preferential (e.g. some Boeing 747 have about 435 seats while others have approximately 305 seats—the cargo area of the latter is bigger; McDonnell Douglas MD 11s have about 300 seats). It is also possible that certain flights are forbidden for certain aircrafts as they might need maintenance during the next week.

It turns out that an assignment of the forthcoming week flight legs to the k aircrafts can be obtained by finding k arc-disjoint paths in an acyclic digraph D defined as follows. The vertex set of D is composed by the aircrafts $p_1, \ldots, p_k$ and all pairs (dc_i, dt_i) and (ac_i, at_i) for all flight number i. In addition, D has k extra vertices $p'_1, \ldots, p'_k$. The graph D has four types of arcs:

(*i*) "flight arcs": for each flight i that is not being carried out at Sunday 0:00h there is an arc from (dc_i, dt_i) to (ac_i, at_i);

(*ii*) "flight in the air at Sunday 0:00h arcs": for each aircraft p_j that is in the air at Sunday 0:00h carrying out flight number i there is an arc in D from p_j to (ac_i, at_i) and from (dc_i, dt_i) to p'_j;

(*iii*) "on the ground arcs": for each city c and each two consecutive times t and t', $t < t'$, between departures or arrivals at c we add m parallel arcs from (c, t) to (c, t'), where m is the number of aircrafts on the ground at city c between time t and t';

(*iv*) "on the ground at Sunday 0:00h arcs": if t is the first time on Sunday (last time on Saturday, respectively) of a departure or arrival at city c then D has an arc from p_j to (c, t) (from (c, t) to p'_j, respectively) for each aircraft on the ground at city c at Sunday 0:00h.

By the definition of D one sees that the vertices $p_1, \ldots, p_k$ have outdegree one and indegree zero, $p'_1, \ldots, p'_k$ have indegree one and outdegree zero and all the other vertices have their indegree equals to their outdegree. We also note that in practice the number of maintenance arcs is typically small. Figure 1.1 gives an impression of how graph D may look like. The home city in this example is Amsterdam[1]. The type (*iii*) arcs are not drawn in the figure to avoid overcrowding (the only type (*iii*) arcs in this example are arcs corresponding to aircrafts on the ground at Amsterdam).

Now, for each $j = 1, \ldots, k$ we define a subset A_j of the arc set of D that

[1]Source: KLM timetable March/95–October/95

can be "traversed" by aircraft p_j. The idea is that A_j should contain the arcs corresponding to flights that can be carried out by the aircraft p_j. For instance, if aircraft p_j must undergo maintenance on next Monday then A_j should be the unique set containing the arc corresponding to the "flight" maintenance on Monday. Likewise if aircraft p_j is carrying out flight i at Sunday 0:00h, then A_j should be the unique set containing the arc in D from p_j to (ac_i, at_i).

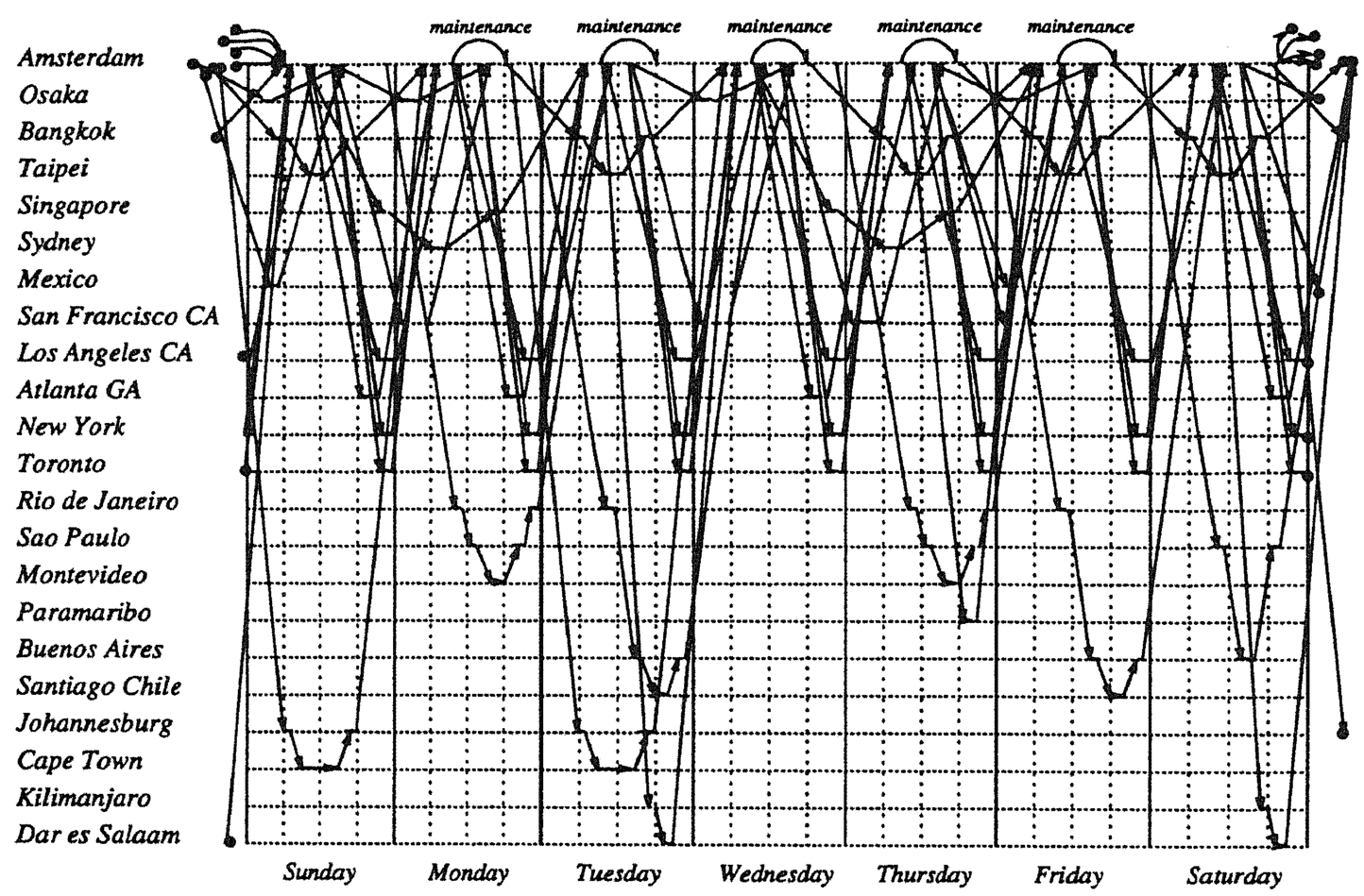


Figure 1.1: An illustration of the acyclic graph corresponding to some flights carried out by KLM's fleet of Boeing 747s.

It follows that to find the wanted assignment of aircrafts to flight legs it suffices to find arc-disjoint paths $P_1, \ldots, P_k$ in D so that P_j is a path in $D[A_j]$ from p_j to any vertex in $\{p'_1, \ldots, p'_k\}$ $(j = 1, \ldots, k)$. So, the begin vertex of each path is fixed while its end vertex can be any one of the vertices in $\{p'_1, \ldots, p'_k\}$; this is due the fact that we do know were each one of the k planes is going to be Sunday 0:00h (either on the ground or performing a flight leg) while it still to be decided were each aircraft will be next week at Saturday 24:00h.

The arc-disjoint paths $P_1, \ldots, P_k$ can be found in polynomial time for each fixed k by using a dynamic programming approach. It is also possible trying to realize the preference for certain aircrafts for carrying out certain flights by defining a "preference" functions $l_j : A_j \to \mathbb{R}_+$ $(j = 1, \ldots, k)$ and seek paths to maximize $\sum_{j=1}^{k} \text{length}_{l_j}(P_j)$, where if $P_j = (v_0, e_1, v_1, \ldots, e_d, v_d)$ then

$\text{length}_{l_j}(P_j) := \sum_{i=1}^{d} l_j(e_i)$ $(j = 1, \ldots, k)$. So, the corresponding assignment maximizes the preference.

In the present chapter we reformulate the above disjoint paths problem in terms of finding a path in a (bigger) directed acyclic graph. This is a known reformulation due to Perl & Shiloach [1978] for finding two vertex-disjoint paths in a directed acyclic digraph and later extended by Fortune, Hopcroft & Wyllie [1980] in order to derive a polynomial-time algorithm for the k-disjoint paths problem in acyclic digraphs (see also Schrijver [1993]). We also discuss the intractability of variants of the disjoint paths problem in acyclic digraphs.

Remark 1.1. The described application of the disjoint paths problem in acyclic graphs to the assignment of aircrafts is due to Schrijver [1993].

1.2. Disjoint paths in acyclic digraphs

Consider the following disjoint paths problem:

$$\begin{array}{ll} \textit{given}: & - \text{ a directed graph } D = (V, A); \\ & - \text{ pairs } (r_1, s_1), \ldots, (r_k, s_k) \text{ of vertices of } D; \\ & - A_1, \ldots, A_k \text{ subsets of } A; \\ & - \text{ a set } H \text{ of pairs } \{i, j\} \text{ from } \{1, \ldots, k\}; \\ \textit{find}: & - \text{ directed paths } P_1, \ldots, P_k \text{ in } D \text{ so that:} \\ & \qquad (i)\ P_i \text{ is an } r_i - s_i\text{-path in } D[A_i]\ (i = 1, \ldots, k); \\ & \qquad (ii)\ P_i \text{ and } P_j \text{ are vertex-disjoint for } \{i, j\} \in H. \end{array} \tag{1.1}$$

Fortune, Hopcroft & Wyllie [1980] showed that this disjoint paths problem is $\mathcal{NP}$-complete even for $k = 2$, $A_1 = \ldots = A_k = A$ and H being the set of all pairs $\{i, j\}$ from $\{1, \ldots, k\}$.

Problem (1.1) is also $\mathcal{NP}$-complete for acyclic graphs (cf. Even, Itai & Shamir [1976]). In fact, the disjoint paths problem (1.1) is $\mathcal{NP}$-complete even for a fixed acyclic digraph, as noted by A. Schrijver. It is easy to see that problem (1.1) (stated as a decision problem) belongs to $\mathcal{NP}$. We transform planar 3-colourability to problem (1.1) restricted to instances having the acyclic digraph D displayed in Figure 1.2 as input. (A *K-colouring* of a graph $G = (V, E)$ is a function $f : V \to \{1, \ldots, K\}$ so that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$. G is *K-colourable* if G has a K-colouring. The *planar 3-colourability problem* is the problem: *given:* a planar graph $G = (V, E)$; *question:* is G 3-colourable? Planar 3-colourability was shown to be $\mathcal{NP}$-complete by Stockmeyer [1973].)

Let us be given a planar graph $G = (V, E)$ with $V = \{v_1, \ldots, v_k\}$ and let f' be a 4-colouring of G (such a function f' can be found in polynomial time (see, e.g., Nishizeki & Chiba [1988: Ch. 5])). We construct an instance of problem (1.1) depending on G and f' as follows. Let $H := \{\ \{i, j\} \mid \{v_i, v_j\} \in E\}$ and for $i = 1, \ldots, k$ let $r_i := u_{f'(v_i)}$, $s_i := w_{f'(v_i)}$ and $A_i := A$. We claim now that G is 3-colourable if and only if the constructed instance of problem

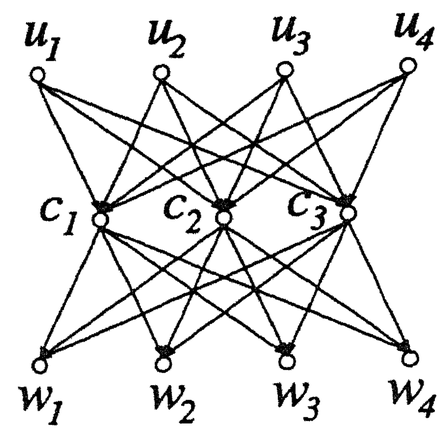


Figure 1.2: Problem (1.1) is $\mathcal{NP}$-complete for instances having this acyclic digraph as input.

(1.1) is feasible. Suppose G is 3-colourable and let f be a 3-colouring of G. For $i = 1, \ldots, k$ let P_i be the path from r_i to s_i that traverses $c_{f(v_i)}$. One can easily check that $P_1, \ldots, P_k$ is a solution to the problem (1.1). Conversely, let $P_1, \ldots, P_k$ be a solution to the disjoint paths problem (1.1) and for $i = 1, \ldots, k$ define $f : V \to \{1, 2, 3\}$ so that the path P_i from r_i to s_i traverses the vertex $c_{f(v_i)}$. It easy to see that f is a 3-colouring of G. The following theorem summarizes our discussion.

Theorem 1.1 *The disjoint paths problem (1.1) restricted to instances having the graph in Figure 1.2 as input is $\mathcal{NP}$-complete.* ∎

Despite the intractability of problem (1.1) even for a fixed acyclic digraph it has been shown by Fortune, Hopcroft & Wyllie [1980], extending the algorithm of Perl & Shiloach [1978] for finding two vertex-disjoint paths in a directed acyclic graph, that for each fixed k the problem of finding k vertex-disjoint paths in an acyclic digraph is polynomially solvable. We prove a generalization of their result, namely we prove that the disjoint paths problem (1.1) is polynomially solvable for instances satisfying the following condition:

There exists no directed cycle $C = P_{j_0} \cdot P_{j_1} \cdot \cdots \cdot P_{j_t}$ in D so that:

$$\begin{aligned} &(i)\ P_{j_i} \text{ is a path from } u_i \text{ to } u_{i+1} \text{ in } D[A_{j_i}], \\ &\quad u_i \neq s_{j_i}\ (i = 0, \ldots, t), \text{ where } u_{t+1} = u_0; \\ &(ii)\ \{j_0, j_1\}, \ldots, \{j_{t-1}, j_t\}, \{j_t, j_0\} \in H. \end{aligned} \tag{1.2}$$

(Recall that if P and Q are paths then $P \cdot Q$ denotes the path obtained by the concatenation of P and Q.) It is clear that any acyclic digraph satisfies the condition above.

Theorem 1.2 *For each fixed k there exists a polynomial-time algorithm for the disjoint paths problem (1.1) for instances satisfying (1.2).*

Proof. Make an auxiliary digraph $D' = (V', A')$ as follows. The vertex set V' consists of all k-tuples $(v_1, \ldots, v_k)$ of vertices of D so that $v_i \neq v_j$ for all $\{i, j\} \in H$. In D' there is an arc from $(v_1, \ldots, v_k)$ to $(w_1, \ldots, w_k)$ if and only if there exists an $i \in \{1, \ldots, k\}$ such that:

$$\begin{aligned} &(i)\ v_j = w_j \text{ for all } j \neq i; \\ &(ii)\ (v_i, w_i) \text{ is an arc of } A_i; \\ &(iii)\ \text{if } j \neq i,\ \{i, j\} \in H \text{ and } v_j \neq s_j \text{ there is no} \\ &\qquad \text{directed path in } D[A_j] \text{ from } v_j \text{ to } v_i. \end{aligned} \tag{1.3}$$

The following holds:

$$\begin{aligned} &D \text{ contains paths } P_1, \ldots, P_k \text{ such that } P_i \text{ is an } r_i - s_i\text{-path} \\ &\text{in } D[A_i]\ (i = 1, \ldots, k) \text{ and } P_i \text{ and } P_j \text{ are vertex disjoint for} \\ &\{i, j\} \in H \\ &\qquad\qquad \text{if and only if} \\ &D' \text{ contains a directed path } P \text{ from } (r_1, \ldots, r_k) \text{ to } (s_1, \ldots, s_k). \end{aligned} \tag{1.4}$$

Suppose that $P_1, \ldots, P_k$ exist. For any i, let P_i follow the vertices $v_{i,0}, v_{i,1}, \ldots, v_{i,t_i}$. So $v_{i,0} = r_i$ and $v_{i,t_i} = s_i$ for each i. Choose $j_1, \ldots, j_k$ such that $0 \leq j_i \leq t_i$ for each i and such that:

$$\begin{aligned} &(i)\ D' \text{ contains a directed path from } (r_1, \ldots, r_k) \text{ to } (v_{1,j_1}, \ldots, v_{k,j_k}); \\ &(ii)\ j_1 + \cdots + j_k \text{ is as large as possible.} \end{aligned}$$

Let $I := \{i \mid j_i < t_i\}$. If $I = \emptyset$ we are done, so assume $I \neq \emptyset$. Then by the definition of D' and the maximality of $j_1 + \cdots + j_k$ there exists for each $i \in I$ an $i' \neq i$ such that there is a directed path in $D[A_{i'}]$ from $v_{i',j_{i'}}$ to v_{i,j_i}, $v_{i',j_{i'}} \neq s_{i'}$ and $\{i', i\} \in H$. So, each point $v_{i,j_i}, i \in I$ is an endpoint of a directed path in $D[A_{i'}]$ starting in another point $v_{i',j_{i'}} \neq s_{j_{i'}}$, $i' \in I$ and $\{i, i'\} \in H$. This contradicts (1.2).

Conversely, let P be a path from $(r_1, \ldots, r_k)$ to $(s_1, \ldots, s_k)$ in D'. Let P follow the vertices $(v_{1,j}, \ldots, v_{k,j})$ for $j = 0, \ldots, t$. So $v_{i,0} = r_i$ for $i = 1, \ldots, k$. For each $i = 1, \ldots, k$ let P_i be path in D following $v_{i,j}$ for $j = 0, \ldots, t$, taking repeated vertices only once. So P_i is a directed $r_i - s_i$-path in $D[A_i]$. Moreover, P_i and P_j are disjoint for each $\{i, j\} \in H$. For suppose P_1 and P_2 (say) have a vertex in common, where $\{1, 2\} \in H$. That is $v_{1,j} = v_{2,j'}$ for some $j \neq j'$. Without loss of generality, $j < j'$ and $v_{1,j} \neq v_{1,j+1}$. By the definition of D', there is no directed path in $D[A_2]$ from $v_{2,j}$ to $v_{1,j}$. This however contradicts the fact that $v_{1,j} = v_{2,j'}$ and that there exists a directed path in $D[A_2]$ from $v_{2,j}$ to $v_{2,j'}$. ∎

The arc-disjoint version of the disjoint paths problem (1.1) consists of replacing (ii) in (1.1) by:

$$P_i \text{ and } P_j \text{ are arc-disjoint for } \{i, j\} \in H. \tag{1.5}$$

This arc-disjoint path problem can be reformulated in terms of the disjoint path problem (1.1). Indeed, let an instance of the arc-disjoint paths problem be given, that is a digraph $D = (V, A)$ pairs of vertices $(r_1, s_1), \ldots, (r_k, s_k)$, arc sets $A_1, \ldots, A_k$ and a set H of pairs $\{i, j\}$ from $\{1, \ldots, k\}$.

We may assume that each r_i is the tail of a unique arc a_i of D and that s_i is the head of a unique arc b_i of D $(i = 1, \ldots, k)$. We make a digraph $D' = (V', A')$ as follows. The vertex set of D' is the arc set A of D (i.e. $V' := A$). In D' there is an arc from a to b if the head of a and the tail of b coincide. For $i = 1, \ldots, k$ we define $A'_i := \{(a, b) \in A' \mid a, b \in A_i\}$. Finally we take $H' := H$.

Finding paths $P_1, \ldots, P_k$ in D satisfying (1.5) so that P_i is an $r_i - s_i$-path in $D[A_i]$ $(i = 1, \ldots, k)$ is equivalent to the problem of finding paths $P'_1, \ldots, P'_k$ in D' satisfying (ii) of (1.1) so that P'_i is $a_i - b_i$-path in $D'[A'_i]$ $(i = 1, \ldots, k)$. Hence, the arc-disjoint version of problem (1.1) is polynomially solvable for instances satisfying a condition similar to condition (1.2).

Remark 1.2. Let us be given length functions $l_i : A_i \to \mathbb{R}_+$. For each fixed k one can also find in polynomial time paths $P_1, \ldots, P_k$ solution to the disjoint path problem (1.1) so that $\sum_{i=1}^{k} \text{length}_{l_i}(P_i)$ is as small as possible (as large as possible, respectively). (Define a length function on the arcs of the auxiliary digraph D' of Theorem 1.2 as follows. The length of an arc of D' from $(v_1, \ldots, v_k)$ to $(w_1, \ldots, w_k)$ satisfying (1.3) is $\sum_{i=1}^{k} l_i((v_i, w_i))$. Now, a shortest (longest, respectively) path from $(r_1, \ldots, r_k)$ to $(s_1, \ldots, s_k)$ in D' with this length function on its arcs gives the desired paths—the *longest path problem* in acyclic graphs can be solved in polynomial time (see, e.g., Lawler [1976: Ch. 3]).)

In the introduction of this chapter the problem of assign k aircrafts $p_1, \ldots, p_k$ to the forthcoming week flight legs of an airline company was reformulated in terms of finding k arc-disjoint paths $P_1, \ldots, P_k$ in an acyclic digraph D, where for $i = 1, \ldots, k$ P_i is a path starting in a given vertex (namely, p_i) and ending in a given set of vertices (namely, $\{p'_1, \ldots, p'_k\}$). The technique above shows that this disjoint paths problem can be solved in polynomial time for each fixed k. (To see this we add to D two new vertices r, s and new arcs (r, p_i), (p'_i, s) $(i = 1, \ldots, k)$. We define $r_i := r$ and $s_i := s$ $(i = 1, \ldots, k)$. Now, it suffices to find k arc-disjoint $r - s$-paths in the new digraph.)

Let us suppose that all the aircrafts $p_1, \ldots, p_n$ are of one type. It is noteworthy that with this assumption the aircraft assignment problem is also polynomially solvable if the number of aircrafts that must undergo maintenance the coming week is fixed (this number seems to be typically small (say 5) in relation to the number k of aircrafts). Let $p_{i_1}, \ldots, p_{i_t}$ be the aircrafts that should be home for maintenance the coming week. Let m_j be the arc of the digraph D representing the maintenance of aircraft p_{i_j} $(j = 1, \ldots, t)$. It turns out that to solve the aircraft assignment problem it suffices to find arc-disjoint paths $P_{i_1}, \ldots, P_{i_t}$ in D so that path P_{i_j} starts at p_{i_j}, traverses the arc m_j

and has its endpoint in $\{p'_1, \ldots, p'_k\}$ $(j = 1, \ldots, t)$. By the indegree=outdegree property of D theses paths can be extended (greedily) to the desired arc-disjoint paths $P_1, \ldots, P_k$ in D. The paths $P_{i_1}, \ldots, P_{i_t}$ can be found in polynomial time for fixed t.

1.3. Further applications: work force scheduling

The disjoint paths problem in acyclic digraphs is fairly generic. Consequently, it arises naturally in many different contexts. Algorithms for finding arc-disjoint paths in acyclic digraphs may also be applicable in the setting of *work force scheduling* problems (see Ahuja, Magnanti & Orlin [1993: Ch. 16]).

In work force scheduling the employees of a company (e.g. airlines crew, nurses, telephone operators,...) should be assigned to carry out some pre-specified services. Typically, the scheduling is complicated by some side constraints as: employees have limits on the number of work hours per day or week; after (say) 5 consecutive working days an employee must have two days off; certain employees might be more suitable than others to carry out certain shifts.

As another illustration of the above technique, suppose we want to schedule k nurses $n_1, \ldots, n_k$ of a hospital to fulfill its eight-hour shifts for the next week (from Sunday 0:00h till Saturday 24:00h). Let us also suppose that: the shifts begin only at full hours; no employee should be assign to work more than (say) 48 hours a week; and that the time off between any two shifts is at least 14 hours. Each shift is given by a begin day (e.g. Saturday) and begin hour (e.g. 18:00h). We add to the end day and end hour of each shift the 14 hours of time off that the employee performing that shift should take (analogously to the aircraft assignment). For example, the end day and hour of the shift starting at Saturday 18:00h is Sunday 16:00h. We make an acyclic digraph $D = (V, A)$ having as a vertex the pair (d, h) for each day d of the week and for each hour h of the day. In addition, D has two vertices n_i and n'_i associated with each nurse. The definition of the arc set of D is similar to the arc set of the aircraft assignment described, that is the digraph D has: "shift arcs" corresponding to shifts during the week; "shift at Sunday 0:00h arcs" corresponding to shifts that start on Saturday and end on Sunday; "time off arcs" corresponding to nurses which are not working during a certain period of the week; and "time off at Sunday 0:00h arcs" corresponding to nurses which are not working at Sunday 0:00h. For each nurse n_i its corresponding arc set A_i can be used to model the shifts that can be performed by this nurse or the shifts that this nurse is willing to carry out. We also define a function $l : A \to \mathbb{Z}_+$ so that $l(a)$ is the number of work hours corresponding to arc a. For instance, for a "time off arc" a we have $l(a) = 0$ and for a "shift arc" a we have $l(a) = 8$. The desired scheduling can be obtained by finding arc-disjoint paths $P_1, \ldots, P_k$ in D so that P_i is a path in $D[A_i]$ and $\text{length}_l(P_i) \leq 48$ $(i = 1, \ldots, k)$. These length-bound disjoint paths can be found in polynomial time for each fixed k (as the number of hours in a week is a constant) by combining the technique

discussed in this chapter with the technique used in Section 5.3.

Chapter 2

VLSI-routing and disjoint paths in planar graphs

2.1. Introduction

The one-layer routing problem in the context of the design of printed circuit boards or VLSI chips can be mathematically formulated as the problem of finding vertex-disjoint paths in a planar graph as follows. Given are the channels on the layer over which wires can be routed and pairs of terminals, called *nets*, which should be equipotential, that is they should be electrically connected by a wire. The terminals are located along the boundary of the layer and along those of certain "modules", i.e. predesigned circuit components. The nets must be interconnected so that wires connecting different nets should not touch each other. The wire routing channels can be modeled by a planar graph G, where modules correspond to faces $I_1, \ldots, I_p$ of G and nets to pairs of vertices $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ along the boundary of the faces $I_1, \ldots, I_p$. The wires connecting the nets correspond to vertex-disjoint $r_i - s_i$-paths in G $(i = 1, \ldots, k)$ (see Figure 2.1).

Edge- or vertex-disjoint paths problems in planar graphs are in general $\mathcal{NP}$-hard (cf. Lynch [1975]), even if the graph together with edges $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ is planar (cf. Middendorf & Pfeiffer [1993]). In some special cases, however, efficient algorithms for the problem are known.

Pinter [1983] proposed a two-phase approach for the one-layer routing problem. In the first phase a sketch of the interconnections between the nets is made. The idea of the sketch is specify the placement of the wires interconnecting the nets with respect to the modules. Figure 2.2(a) illustrates a sketch. (In the figure, the thicker lines interconnecting the nets may be given by paths in the grid graph.)

Next, in the second phase one tries to route the nets according to the

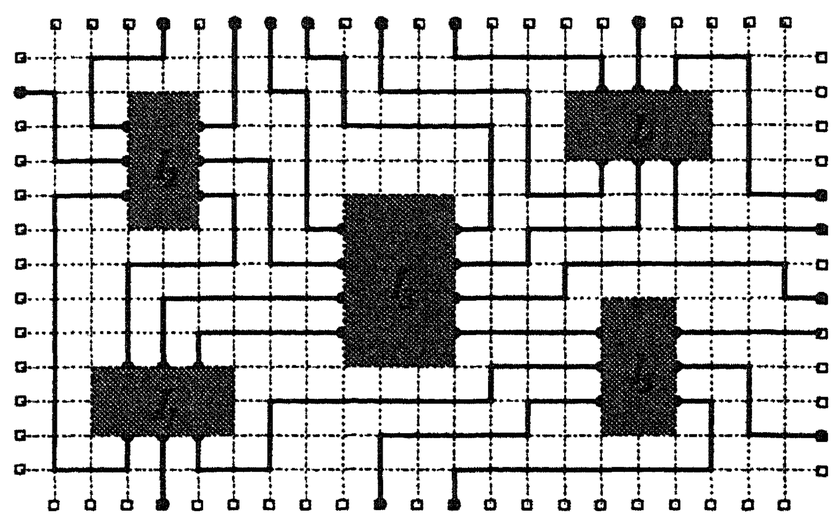


Figure 2.1: An illustration of an one-layer routing. The gray area represent the modules, the dotted grid indicates the routes a wire can potentially make and the thicker lines represent the wires. The nets are shown in the figure as black dots.

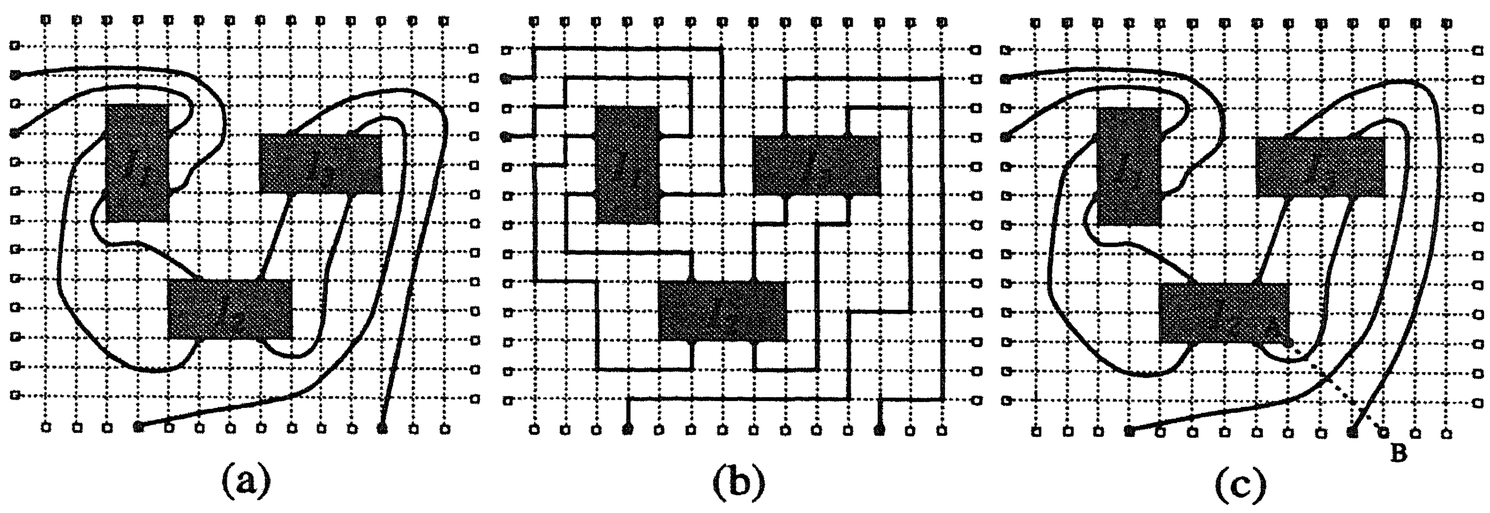


Figure 2.2: Figure (a) shows a sketch of the interconnections between the nets. Figure (b) is a routing corresponding to the sketch in (a). Figure (c) displays an infeasible sketch.

prescribed sketch. Figure 2.2(b) shows a routing of the one-layer sketched in Figure 2.2(a).

A sketch need not be feasible. Figure 2.2(c) shows an example of a sketch that cannot be routed. (It is also possible that the one-layer instance itself is infeasible.) Cole & Siegel [1984] have given a "cut-condition" characterizing the feasibility of a sketch. For instance, the line between points A and B in Figure 2.2(c) shows a "oversaturated" cut—the channel between the two points is too narrow for the wires to be routed.

As we mentioned, the one-layer routing problem is $\mathcal{NP}$-complete. Leiserson & Maley [1985] have shown that the second phase of Pinter's approach is polynomially solvable for grid graphs; a very important class of graphs for practical proposes (the graph associated to wire routing channels is typically a grid). This indicates that the really intractable step is to "guess" a feasible sketch (if there exists any).

It was shown by Schrijver [1991] that the second phase of Pinter's approach is also polynomially solvable for planar graphs. Let us denote the set of faces of a planar graph G embedded in $\mathbb{R}^2$ by $\mathcal{F}_G$. More specifically, Schrijver presented a polynomial-time algorithm to solve the following *disjoint homotopic paths problem*:

$$\begin{array}{ll} \textit{given}: & \text{– a planar graph } G=(V,E) \text{ embedded in the plane } \mathbb{R}^2; \\ & \text{– a subset } \{I_1,\ldots,I_p\} \text{ of the faces of } G; \\ & \text{– paths } Q_1,\ldots,Q_k \text{ in } G, \text{ each with endpoints on the} \\ & \quad \text{boundary of } I_1 \cup \ldots \cup I_p; \\ \textit{find}: & \text{– pairwise disjoint simple paths } P_1,\ldots,P_k \text{ in } G \text{ so that,} \\ & \quad \text{for each } i=1,\ldots,k,\ Q_i \text{ can be shifted to } P_i \text{ by} \\ & \quad \mathcal{F}_G \setminus \{I_1,\ldots,I_p\}. \end{array} \tag{2.1}$$

We explain the terminology used here. We do not make any distinction between a face of G and the cycle of G bounding it.

We say that a path Q can be *shifted* to a path Q' by $\mathcal{F}_G \setminus \{I_1,\ldots,I_p\}$ (or simply, Q can be shifted to Q') if there exist paths $Q=Q_1,Q_2,\ldots,Q_{t+1}=Q'$ so that for $i=1,\ldots,t$ there is a $F_i \in \mathcal{F}_G \setminus \{I_1,\ldots,I_p\}$ and there are paths R_i and S_i such that:

$$\begin{aligned} &(i)\ F_i = P_i \cdot (P_i')^{-1}; \\ &(ii)\ Q_i = R_i \cdot P_i \cdot S_i; \\ &(iii)\ Q_{i+1} = R_i \cdot P_i' \cdot S_i. \end{aligned}$$

If Q can be shifted to Q' by $\mathcal{F} \setminus \{I_1,\ldots,I_p\}$ we also say that Q and Q' are *homotopic* in $\mathbb{R}^2 \setminus \{I_1,\ldots,I_p\}$, or just *homotopic* (See Figure 2.3.) The relation "be homotopic" defines an equivalence relation on the set of paths in G. One may think of homotopic paths as elastics with fixed endpoints that can be

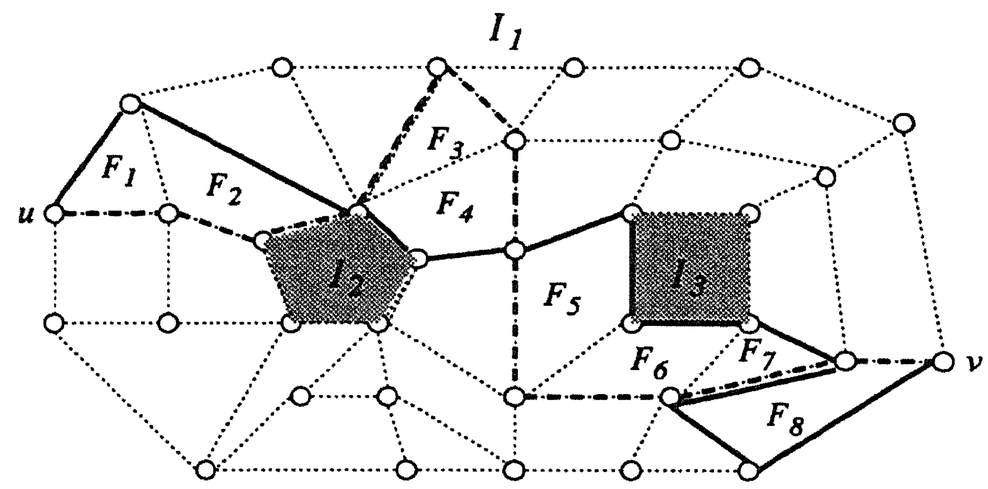


Figure 2.3: The figure shows two homotopic paths – the one drawn by dashed lines and the one drawn by solid lines.

deformed, stretched and shifted to each other as long as the boundaries of the faces $I_1, \ldots, I_p$ are not crossed.

Schrijver has reduced problem (2.1) to the problem of solving a system $Ax \leq b$ in integers, where A is a $\{0, \pm 1\}$-matrix with at most two nonzero entries per row. An integral solution of $Ax \leq b$ is then used to shift the paths $Q_1, \ldots, Q_k$. In this chapter we describe how the system $Ax \leq b$ is constructed and the paths are shifted. In the next chapter we present a fast method to solve $Ax \leq b$ in integers. Before proceeding we need a few definitions.

2.2. Some definitions and assumptions

Without loss of generality we may assume that: (i) I_1 is the unbounded face of G; (ii) G is 2-connected; (iii) the endpoints of each Q_i have degree one in G; (iv) each edge of G is traversed by at most one Q_i; no edge traversed by any Q_i, except for the first and last edge of Q_i, is incident with any face in $\{I_1, \ldots, I_p\}$. Conditions (iii) and (iv) can be achieved by adding new vertices and (parallel) edges.

Let H be the graph with lengths on its edges obtained from G as follows. Let $V_{\mathcal{F}_G} := \{v_F \mid F \notin \mathcal{F}_G \setminus \{I_1, \ldots, I_p\}\}$. The vertex set of H is $V \cup V_{\mathcal{F}_G}$ and each edge of H is of the form (v, v_F) where $v \in V$, $v_F \in V_{\mathcal{F}_G}$, and the vertex v is incident to the face F in G. The length of each edge of H is $1/2$. So, it is clear that H is a bipartite planar graph and that it can be embedded in $\mathbb{R}^2$ so that $G + H$ is a planar embedding. Each face I_i of G corresponds naturally to a face I_i' of H $(i = 1, \ldots, p)$ (see Figure 2.4).

Let for each $i = 1, \ldots, k$

$$Q_i = (v_{i,0}, e_{i,1}, v_{i,1}, \ldots, e_{i,m_i}, v_{i,m_i}). \tag{2.2}$$

We see each Q_i as a directed path from $v_{i,0}$ to v_{i,m_i} (so, we orientate the paths arbitrary) and define the right- and the left-hand side of Q_i according to this

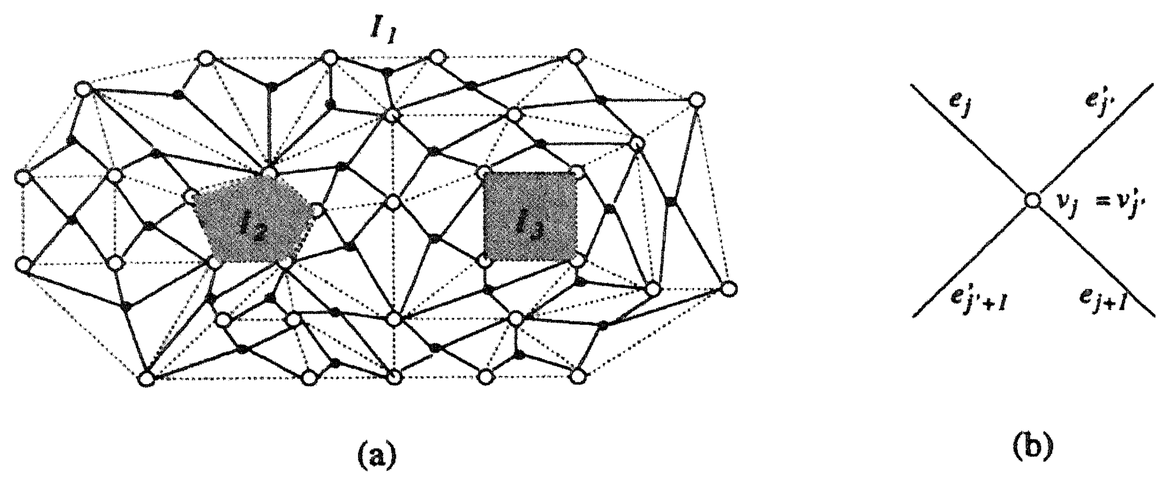


Figure 2.4: Figure (a) shows the graph H (in solid lines) corresponding to the graph G of Figure 2.3. Figure (b) shows two crossing paths.

orientation.

Two paths

$$Q = (v_0, e_1, v_1, e_2, \ldots, e_m, v_m) \text{ and } Q' = (v'_0, e'_1, v'_1, e'_2, \ldots, e'_{m'}, v'_{m'}).$$

in G or H are *crossing* if there exist a pair (j, j') with $1 \leq j \leq m-1$ and $1 \leq j' \leq m'-1$ so that $v_j = v'_{j'}$ and the edges $e_j, e'_{j'}, e_{j+1}, e'_{j'+1}$ occur in this order cyclically at v_j (say) clockwise (see Figure 2.4(b)). In addition, if $Q = Q'$, $v_0 = v'_0$ and $j \neq j'$ then we say that Q is *self-crossing*.

One of the basic steps of Schrijver's algorithms to solve (2.1) consists of modifying $Q_1, \ldots, Q_k$ so that they do not have (self-)crossings or subpaths homotopic to a vertex (i.e. a subpath $(v_{i,j}, e_{i,j+1}, \ldots, e_{i,j'}, v_{i,j'} = v_{i,j})$ of Q_i which is homotopic to $(v_{i,j})$. In the sequel we assume that $Q_1, \ldots, Q_k$ have been preprocessed by Schrijver's "uncrossing" procedure and do not have (self-)crossings or subpaths homotopic to a vertex. (It is clear that if $Q_1, \ldots, Q_k$ cannot be "uncrossed" then this particular instance of problem (2.1) is infeasible.)

2.3. Constructing the system $Ax \leq b$

We introduce a variable $x_{i,j}$ for each $i = 1, \ldots, k$ and $j = 1, \ldots, m_i - 1$. We put a number of linear constraints on $x_{i,j}$ so that any integer solution gives us a solution to (2.1). So, for each vertex of G, each time it is traversed by some Q_i a variable is introduced indicating how far Q_i should be shifted in order to make the paths simple and disjoint.

The meaning associate to each variable $x_{i,j}$ is as follows. If in a solution of the system the value of the variable $x_{i,j}$ is $r \in \mathbb{Z}$, then at vertex $v_{i,j}$ the path Q_i will be shifted over a "distance" of $|r|$ to the right if r is positive or to the left if r is negative (see Figure 2.5(a)).

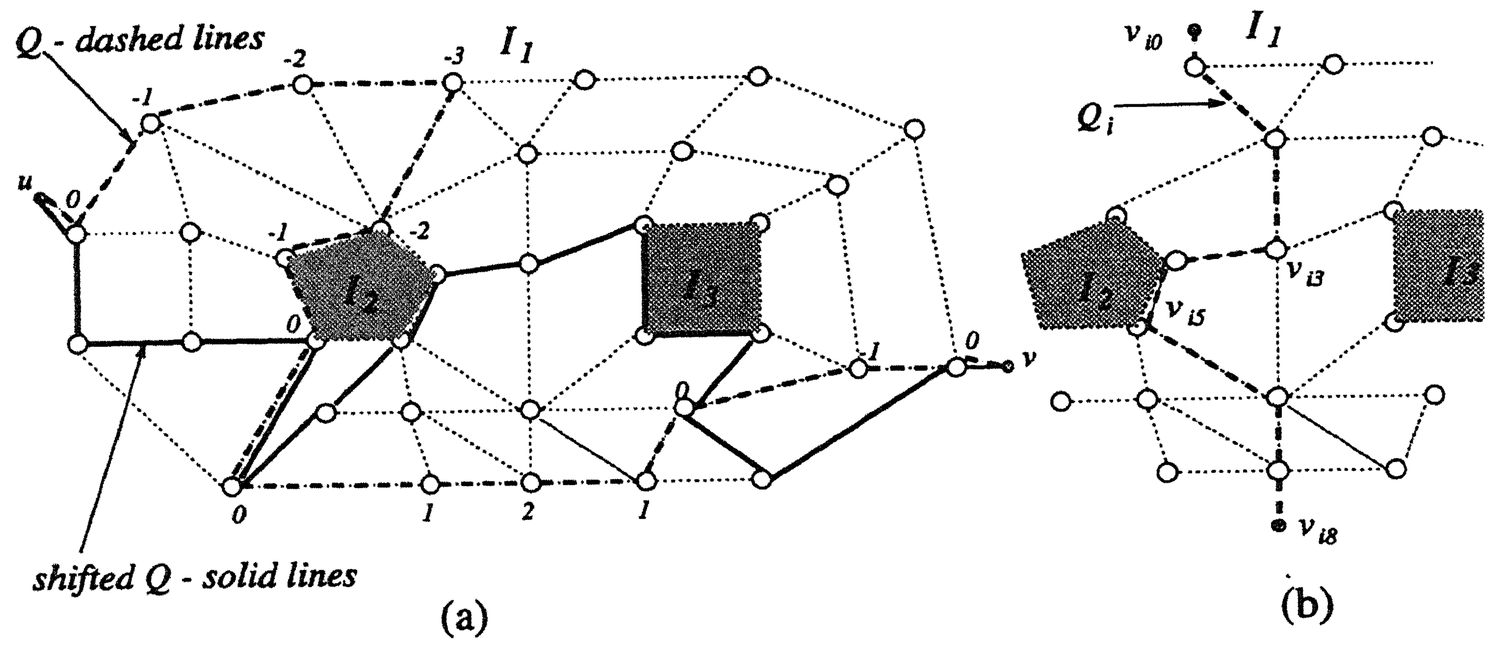


Figure 2.5: Figure (a) gives an idea of how an integral solution of $Ax \leq b$ is used to shift the paths $Q_1, \ldots, Q_k$. The number next to a vertex represents how far and in which direction the path Q must be shifted at this particular vertex. Some of the class 1 inequalities derived from path Q_i of Figure (b) are: $x_{i,3} \leq 1$, $-x_{i,3} \leq 1$, $-x_{i,5} \leq 2$ and $x_{i,5} \leq 0$.

We now describe the three classes of inequalities forming the system $Ax \leq b$. We shall denote by $e_{i,j,F}$ the edge of H connecting the vertex $v_{i,j} \in V$ to the vertex $v_F \in V_{\mathcal{F}_G}$.

Class 1. This class of inequalities is intended to avoid that a path Q_i is shifted at $v_{i,j}$ over any of the faces $I_1, \ldots, I_p$. For each $i = 1, \ldots, k$ and $j = 1, \ldots, m_i - 1$, we require:

$$\begin{aligned} x_{i,j} &\leq \min_P \text{length}_H(P); \\ -x_{i,j} &\leq \min_P \text{length}_H(P). \end{aligned}$$

Here, the minimum in the first inequality ranges over all paths P in H connecting $v_{i,j}$ to a vertex w incident to a face in $\{I'_1, \ldots, I'_p\}$ so that:

(i) P does not cross any one of the paths $Q_1, \ldots, Q_k$;
(ii) either a face in $\{I'_1, \ldots, I'_p\}$ on the right-hand side of Q_i is incident to $v_{i,j}$ or $P = (v_{i,j}, e_{i,j,F}, v_F, \ldots, w)$ where F is a face on the right-hand side of Q_i.

Likewise, the minimum in the second inequality ranges over all paths P in H connecting $v_{i,j}$ to a vertex w incident to a face in $\{I'_1, \ldots, I'_p\}$ so that P satisfies (i) above and satisfies (ii) after replacing both "on the right-hand side" by "on the left-hand side". See Figure 2.5(b).

Class 2. This class of inequalities guarantees that after two distinct paths Q_i

and Q'_i are shifted at $v_{i,j}$ and $v_{i',j'}$, respectively, then the shifted paths do not intersect. Thus, for each $i, i' = 1, \ldots, k$ with $i \neq i'$, $j, j' = 1, \ldots, m_i - 1$ we require:

$$\begin{aligned} x_{i,j} + x_{i',j'} &\leq \min_P \text{length}_H(P) - 1, \\ x_{i,j} - x_{i',j'} &\leq \min_P \text{length}_H(P) - 1, \\ -x_{i,j} - x_{i',j'} &\leq \min_P \text{length}_H(P) - 1, \end{aligned}$$

where the minimum ranges over all $v_{i,j} - v_{i',j'}$-paths P so that:

(*i*) P does not cross any one of the paths $Q_1, \ldots, Q_k$.

(*ii*) either $v_{i,j} = v_{i',j'}$ or $P = (v_{i,j}, e_{i,j,F}, v_F, \ldots, v_{F'}, e_{i',j',F'}, v_{i',j'})$ where F is a face on the right-hand side of Q_i if the coefficient of $x_{i,j}$ is 1 and F is a face on the left-hand side of Q_i if the coefficient of $x_{i,j}$ is -1 and similarly with respect to F' and the coefficient of $x_{i',j'}$.

See Figure 2.6. The right-hand side of these inequalities may be infinite.

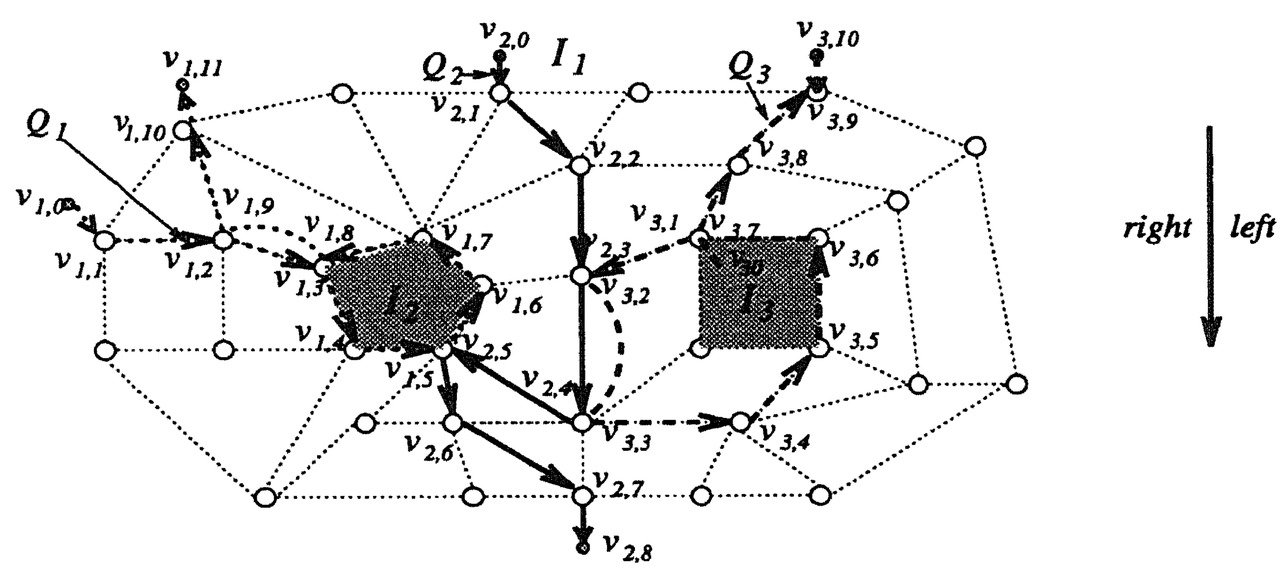


Figure 2.6: Some of the Class 2 inequalities derived from Q_1, Q_2, Q_3 in the figure are: $x_{1,5} + x_{2,5} \leq -1$; $-x_{2,3} + x_{3,2} \leq -1$; $-x_{2,4} + x_{3,3} \leq -1$; $x_{1,4} + x_{1,5} \leq 0$; $x_{1,6} + x_{2,2} \leq 0$; $-x_{2,5} + x_{3,4} \leq 2$; $-x_{2,6} + x_{3,3} \leq 0$; and $-x_{2,2} - x_{3,8} \leq 0$. Some of the Class 3 inequalities are: $-x_{1,2} + x_{1,9} \leq -1$, $x_{3,1} - x_{3,7} \leq -1$ and $x_{2,3} - x_{3,8} \leq 0$.

Class 3. This last class of inequalities enforces that if Q_i is shifted at $v_{i,j}$ and at $v_{i,j'}$ then the shifted Q_i is a simple path. Thus, for $i = 1, \ldots, k$ and $j, j' = 1, \ldots, m_i - 1$, we require:

$$\begin{aligned} x_{i,j} + x_{i,j'} &\leq \min_P \text{length}_H(P) - 1, \\ x_{i,j} - x_{i,j'} &\leq \min_P \text{length}_H(P) - 1, \end{aligned}$$

$$-x_{i,j} - x_{i,j'} \quad \leq \quad \min_P \text{length}_H(P) - 1,$$

where P ranges over all $v_{i,j} - v_{i,j'}$-paths P in H so that:

(*i*) P does not cross any one of the paths $Q_1, \ldots, Q_k$;

(*ii*) $P = (v_{i,j}, e_{i,j,F}, v_F, \ldots, v_{F'}, e_{i',j',F'}, v_{i',j'})$ where F is a face on the right-hand side of Q_i if the coefficient of $x_{i,j}$ is 1 and F is a face on the left-hand side of Q_i if the coefficient of $x_{i,j}$ is -1 and similarly with respect to F' and the coefficient of $x_{i',j'}$;

(*iii*) P is not homotopic to the path Q connecting $v_{i,j}$ to $v_{i,j'}$ which traverses only vertices associated to faces on the right-hand side of Q_i and vertices in $\{v_{i,j}, v_{i,j+1}, \ldots, v_{i,j'}\}$. (So Q is "parallel" to the subpath of Q_i connecting $v_{i,j}$ to $v_{i,j'}$.)

See Figure 2.6. The right-hand side of these inequalities may be infinite.

The system of linear inequalities $Ax \leq b$ formed by the Class 1, 2 and 3 inequalities is so that A is a $\{0, \pm 1\}$-matrix with at most two nonzero entries per row and it can be constructed in polynomial time. Indeed, the shortest paths necessary to write the inequalities in the Class 1 and 2 can be found by applying breadth-first search in the graph H being careful of not crossing any $Q_1, \ldots, Q_k$. Finding a path P that minimizes the right-hand side of the inequalities in the Class 3 corresponds to the *shortest nonhomotopic path problem*, that is:

$$\begin{array}{ll} \textit{given}: & \text{– a planar graph } H = (V, E) \text{ embedded in the plane } \mathbb{R}^2; \\ & \text{– a subset } I_1, \ldots, I_p \text{ of the faces of } H; \\ & \text{– a path } Q \text{ in } H \text{ connecting } u \text{ and } w; \text{ and,} \\ & \text{– a length function } l : E \to \mathbb{Z}_+; \\ \textit{find}: & \text{– a shortest } u - w\text{-path } P \text{ in } H \text{ so that } P \text{ is not} \\ & \quad \text{homotopic to } Q \text{ in } \mathbb{R}^2 \setminus \{I_1, \ldots, I_p\}. \end{array} \tag{2.3}$$

Schrijver [1991] showed that problem (2.1) can be solved in polynomial time as well. Schrijver proved that the following collection of paths contains a solution to (2.3)[1]:

$$\begin{array}{ll} P_{u,v} \cdot P_{v,w} & (v \in V), \\ P_{u,v} \cdot e \cdot P_{v',w} & (e = \{v, v'\} \in E), \end{array} \tag{2.4}$$

where $P_{x,y}$ denotes a shortest $x - y$-path in G. So, in order to solve the shortest nonhomotopic path problem one has to compute this collection of paths (which can be done in polynomial time) and pick up from it the shortest path nonhomotopic to Q.

[1]The similarity between (2.4), (7.7) and (7.30) is noteworthy.

It turns out that $Ax \leq b$ has an integral solution if and only if the disjoint homotopic paths problem (2.1) is feasible (cf. Schrijver [1991]). Next we describe how Schrijver used an integral solution of $Ax \leq b$ in order to shift the paths $Q_1, \ldots, Q_k$.

Remark 2.1. Let N and M be the number of variables and inequalities of $Ax \leq b$. Since $b_i \geq -1$ $(i = 1, \ldots, N)$ it follows from Theorem 3.3 that $Ax \leq b$ can be solved in $O(\sqrt{N}M)$ time by using the single-source shortest path algorithm due to Goldberg [1993] (see Preliminaries on the shortest path problem). Schrijver's original method for solving $Ax \leq b$ by using Fourier-Motzkin elimination of variables runs in time $O(N^3)$.

2.4. Shifting $Q_1, \ldots, Q_k$

An integer solution $(x_{i,j} \mid i = 1, \ldots, k; j := 1, \ldots, m_i - 1)$ determines how far and in which direction each vertex of each path Q_i should be shifted. Schrijver proposed the following iterative method based on a "local" shift of the paths.

Let $M := \max\{|x_{i,j}| \mid i = 1, \ldots, k; j := 1, \ldots, m_i - 1\}$. If $M = 0$ then $x_{i,j} = 0$ for all i, j, and by the Class 2 inequalities the paths $Q_1, \ldots, Q_k$ are disjoint and by the Class 3 inequalities the paths are simple and we are home. Thus, let us suppose $M > 0$. Without loss of generality we may assume $x_{1,g} = M > 0$, $2 \leq g \leq m_i - 2$. Now, we shift Q_1 at v_{1g} a distance of one to the right obtaining a new path $\tilde{Q}_1$. More precisely, let $F_1, \ldots, F_t$ be faces incident to $v_{1,g}$ on the right-hand side of Q_1. Let $\mathcal{F}'$ be the faces enclosed by the outer boundary of $F_1 \,\Delta\, F_2 \,\Delta\, \cdots \,\Delta\, F_t$. We define

$$\tilde{Q}_1 := Q_1 \,\Delta\, \{F \mid F \in \mathcal{F}'\}.$$

(See Figure 2.7(a)).

By the Class 1 inequalities $I_i \notin \mathcal{F}'$ $(i = 1, \ldots, p)$. So Q_1 and $\tilde{Q}_1$ are homotopic. Let $e_1, \ldots, e_{t-1}$ be the edges on the right-hand side of Q_1 incident to v_{1g} as in Figure 2.7(b). By the maximality of M and by the Class 2 inequalities no path $Q_2, \ldots, Q_k$ traverse any one of the edges $e_1, \ldots, e_{t-1}$. Indeed, suppose by contradiction that there exists an edge e_r, $1 \leq r \leq t-1$, which is traversed by a path Q_i, $i \neq 1$. Take $r := \min\{1, \ldots, t-1\}$. Suppose (say)

$$Q_i = (v_{i,0}, e_{i,0}, v_{i,1}, \ldots, e_{i,j}, v_{i,j} = v_{1,g}, e_{i,j+1} = e_r, \ldots, e_{i,m_i}, v_{i,m_i}).$$

Thus, $x_{1,g} + x_{i,j} \leq -1$ is a Class 2 inequality in $Ax \leq b$ implying $-x_{i,j} \geq x_{1,g} + 1 = M + 1$ contradicting the maximality of M. Analogously, by the Class 3 inequalities and the maximality of M no edge in $\{e_1, \ldots, e_{t-1}\}$ is traversed by Q_1. Hence, $\tilde{Q}_1, Q_2, \ldots Q_k$ do not have (self-)crossings or subpaths homotopic to a vertex and these paths give us new variables $\tilde{x}_{i,j}$. We set them equal to the original $x_{i,j}$ if $i \neq 1$, while $\tilde{x}_{1,j}$ are set to $M - 1$ on the new part of $\tilde{Q}_1$ and equal to the original values on the unchanged part of $\tilde{Q}_1$. Now, one can verify that the vector $\tilde{x}$ as defined forms an integral solution of the system of linear inequalities derived from $\tilde{Q}_1, Q_2, \ldots, Q_k$.

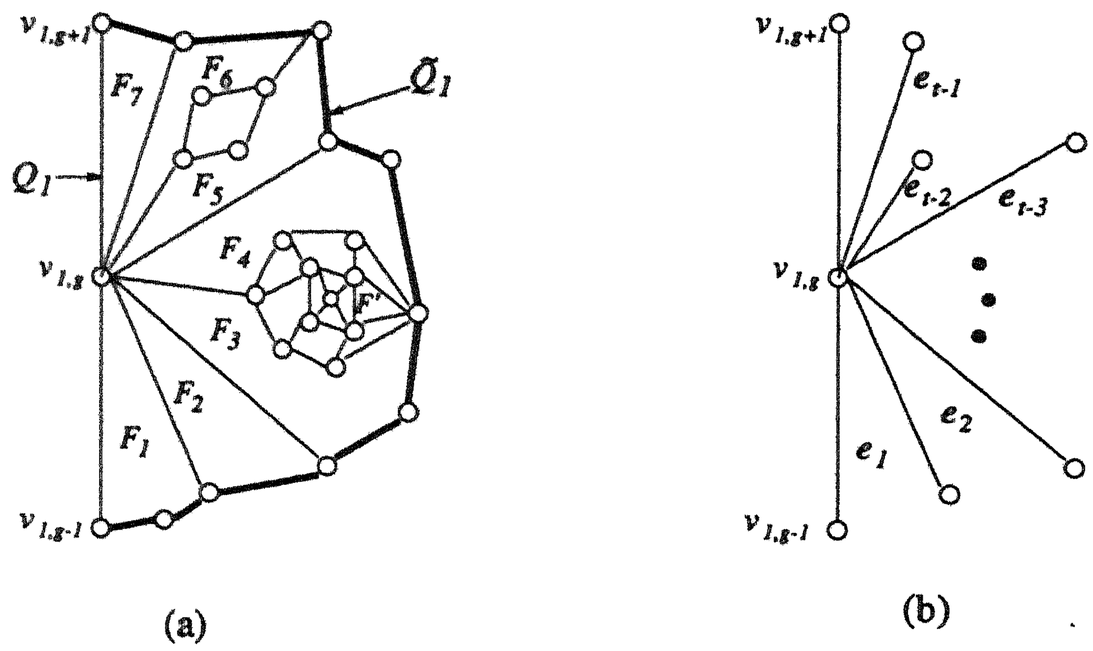


Figure 2.7: Figure (a) shows a local shift. Observe that if $F' \in \{I_1, \ldots, I_p\}$ then $2x_{1,g} \leq 1$ (i.e., $x_{1,g} \leq 0$) would be a Class 3 inequality of $Ax \leq b$.

Thus, each iteration of the above shifting procedure consists of a "local" shift and an updating of variables. It is easy to see that each iteration can be carried out in polynomial time and that the procedure eventually terminates as the number of variable $x_{i,j}$ with $|x_{i,j}| = M$ strictly decreases in each iteration. The number of iterations in the shift procedure is also polynomially bounded. By the Class 1 of inequalities we have that M is bounded by n:= the number of vertices of G. The number of variables $x_{i,j}$ at each stage can be seen to be bounded by nm, where m is the number of edges of G before adding parallel edges to it.

[Let F be a face of G enclosed by a circuit of the form $(v = v_{i,j}, e_{i,j}, w, e_{i',j'}, v)$. Assume (say) $v_{i',j'} = v$ and F is a face incident to $e_{i,j}$ on the right-hand side (consequently, F is incident to $e_{i',j'}$ on the left-hand side). Thus, the inequality $x_{i,j} - x_{i',j'} \leq -1$ is an inequality of the system $Ax \leq b$ (a Class 2 or Class 3 inequality, depending on whether or not $i = i'$). Hence, $x_{i,j} + 1 \leq x_{i',j'} \leq M \leq n$, implying that either the number of parallel edges between any two vertices of G is bounded by n or the present instance of the disjoint homotopic paths problem is infeasible (here we are using that $Q_1, \ldots, Q_k$ do not have subpaths homotopic to a vertex). Hence, the number of variables $x_{i,j}$ at any stage of the algorithm is bounded by nm.]

Therefore, the total number of shift iterations is bounded by n^2m, proving the polynomiality of the whole algorithm.

Chapter 3

Integral potentials

3.1. Introduction

As we have seen in Chapter 2, in one of the steps of Schrijver's algorithm a system $Ax \leq b$ must be solved in integers, where b is an integral m-vector and

$$A = (a_{i,j}) \text{ is an } m \times n - \{0, \pm 1\}\text{-matrix with at most two nonzero entries per row.} \tag{3.1}$$

In the present chapter we focus on this problem, namely on the problem:

$$\begin{array}{rl} \textit{given}: & \text{an integral } m\text{-vector } b \text{ and a matrix } A \text{ satisfying (3.1);} \\ \textit{find}: & \text{an integral solution of } Ax \leq b, \text{ if there exists any.} \end{array} \tag{3.2}$$

Using Fourier-Motzkin elimination of variables Schrijver proposed an algorithm which can be seen to solve (3.2) in $O(n^3)$ time and space (cf. Schrijver [1991]). Moreover, Schrijver characterized the existence of an integral solution. We describe an algorithm for solving (3.2) with the same time and space complexity as the single-source shortest path problem, that is, we show that (3.2) can be solved in $O(T_{ss}(m,n))$ time. (Recall that $T_{ss}(m,n)$ denotes the function expressing the time complexity for solving the single-source shortest path problem in a graph with m arcs and n vertices and possibly negative arc lengths.) Thus, in sparse graphs (i.e. graphs where $m = O(n)$) the algorithm we describe runs in $O(n^2 \log n)$ time (see preliminaries on the shortest path problem), which is faster than Schrijver's original method by a factor of $n/\log n$.

The algorithm we present is based on (and a proof of) the fact that:

(i) if $Ax \leq b$ is feasible then it has a half-integral solution; and
(ii) if $Ax \leq b$ has an integral solution then any half-integral solution of $Ax \leq b$ can be rounded to an integral solution.

Suppose $Ax \leq b$ is feasible and let x^* be a half-integral solution of $Ax \leq b$. We shall transform the problem of rounding x^* to the problem of finding a $\{0,1\}$-solution of a system $Cy \leq d$ where the inequalities have exactly two variables per inequality and for each pair y_j, y_k of variables, any inequality involving y_j and y_k is one of the following:

$$y_j + y_k \leq 1; \quad y_j - y_k \leq 0; \quad y_k - y_j \leq 0; \quad -y_j - y_k \leq -1. \tag{3.3}$$

In Section 3.2 we associate with a system $Ax \leq b$, given in (3.2), a digraph D_A with lengths on its arcs. Next, in Section 3.3 we observe how a half-integral solution of $Ax \leq b$ can be found in $O(T_{ss}(m,n))$ time, provided that $Ax \leq b$ is feasible. In Section 3.4 we present a linear-time algorithm to find a $\{0,1\}$-solution of a system $Cy \leq d$ satisfying the above conditions. In Section 3.5, using the algorithm of Section 3.4, we show that either a half-integral solution of $Ax \leq b$ can be rounded to an integral solution or $Ax \leq b$ has no integral solution at all. It turns out that rounding a half-integral solution of $Ax \leq b$ can be done with the help of linear-time algorithms to: (i) find strongly connected components of a digraph; and (ii) find a topological ordering of the vertices of an acyclic digraph. Finding a half-integral solution of $Ax \leq b$ will be the time-dominating step while the rounding step is realizable in linear time. Finally, this chapter concludes with Section 3.6 discussing the equivalence of the results in Chapter 3 and known results of Even, Itai & Shamir [1976], Schrijver [1978] and Hurkens [1989].

Remark 3.1. The problem of finding a $\{0,1\}$-solution of $Cy \leq d$ can be seen to be equivalent to the *2-satisfiability problem* (satisfiability with at most two literals per clause). Indeed, let $U = \{u_1, \ldots, u_n\}$ be a set of variables and let F be a formula on the set U having a two-literal clause corresponding to each inequality of $Cy \leq d$ according to the following rule:

$$\begin{array}{rcrcrcr} y_j^* + y_k^* & \leq & 1 & \longleftrightarrow & \neg u_j & \vee & \neg u_k \\ y_j^* - y_k^* & \leq & 0 & \longleftrightarrow & \neg u_j & \vee & u_k \\ -y_j^* - y_k^* & \leq & -1 & \longleftrightarrow & u_j & \vee & u_k \end{array} .$$

One can easily check that $Cy \leq d$ has a $\{0,1\}$-solution if and only if there exists a truth assignment $t : U \to \{\textit{true}, \textit{false}\}$ satisfying all the clauses in F.

3.2. The digraph associated with $Ax \leq b$

Let A be a matrix satisfying (3.1). We associate with a system $Ax \leq b$ a digraph $D_{A,b}$ with length on its arcs as follows. (We will write D_A instead of $D_{A,b}$ if no confusion results.) For each column index j of A, D_A has two corresponding vertices, j^+ and j^-. If row i has nonzero in positions j and k with $j \neq k$, then D_A has arcs of length b_i connecting j^+, j^-, k^+, k^-, as shown in Figure 3.1, depending on whether $(a_{i,j}, a_{i,k}) = (1,1), (1,-1), (-1,1)$ or $(-1,-1)$.

Finally, if row i has exactly one nonzero a_{ij} being $+1$ (-1 respectively), D_A has an arc (j^-, j^+) ((j^+, j^-) respectively) of length $2b_i$, as shown in

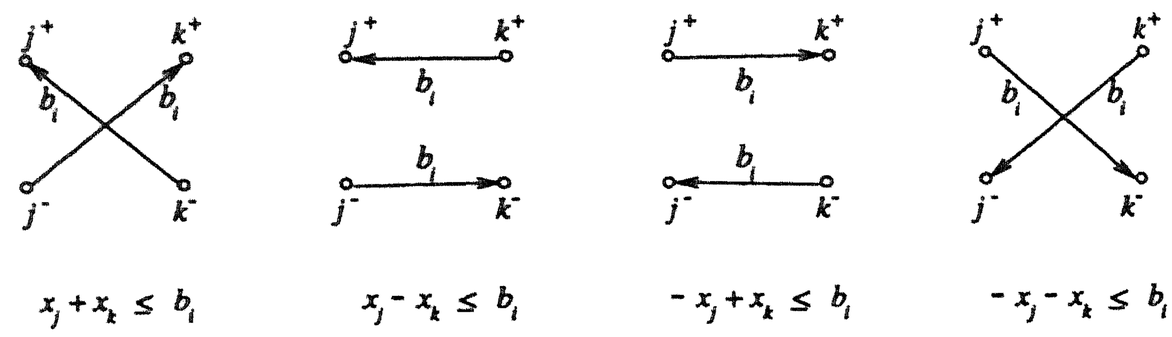


Figure 3.1: Arcs of D_A corresponding to inequalities of the type $\pm x_j \pm x_k \leq b_i$.

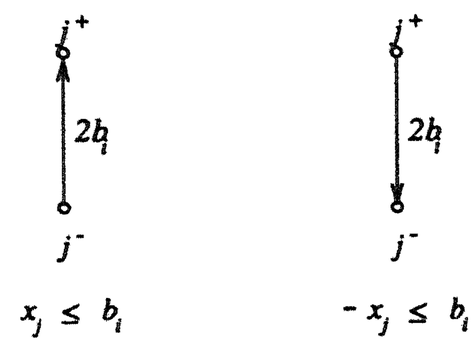


Figure 3.2: Arcs of D_A corresponding to inequalities of the type $\pm x_j \leq b_i$.

Figure 3.2. We shall denote the length of an arc (u, v) of a digraph D by $\text{length}_D(u, v)$.

3.3. Finding a half-integral solution of $Ax \leq b$

We call a function π on the vertices of a graph D satisfying

$$\pi(v) - \pi(u) \leq \text{length}_D(u, v) \qquad \text{for each arc } (u, v) \text{ of } D$$

a *potential function on* D. If x^* is a solution of $Ax \leq b$ and π^* is the function defined as

$$\pi^*(j^+) := x_j^*, \quad \pi^*(j^-) := -x_j^* \quad \text{for each column index } j \text{ of } A, \tag{3.4}$$

then π^* is a potential function on D_A.

Thus, any solution of $Ax \leq b$ gives rise to a potential function on D_A. Conversely, if π^* is a potential function on D_A then x^* defined as

$$x_j^* := (\pi^*(j^+) - \pi^*(j^-))/2 \quad \text{for each column index } j \text{ of } A \tag{3.5}$$

is a solution of $Ax \leq b$. Therefore, there exists a solution of $Ax \leq b$ if and only if there exists a potential function on D_A. Moreover, if a potential function is integral then the corresponding solution of $Ax \leq b$ is half-integral. In order to find a (half-integral) solution of $Ax \leq b$ we find an (integral) potential function on D_A as described below.

A graph D has a potential function if and only if D has no negative-length cycle. As it is well-known, to find a potential function on D we can proceed as follows. First we add to D a new vertex s and zero-length arcs (s, v) for every vertex v in D. Then for each vertex v we compute the length of a shortest path from s to v in the augmented graph, denoted by $\text{distance}_D(s, v)$. There exists a shortest path from s to each vertex if and only if the augmented graph (and consequently D) has no negative-length cycle. If $\text{distance}_D(s, v)$ is well-defined for every vertex v, then $\pi(v) := \text{distance}_D(s, v)$ for each vertex v of D, is a potential function on D.

Hence, with one application of a single-source shortest path algorithm we either find a potential function on D or a negative-length cycle proving that D has no potential function at all. Moreover, if all the arc lengths are integer then the potential function found is integral. We summarize the preceding discussion in the following proposition.

Proposition 3.1 *Let b be an m-vector and let A be a matrix satisfying (3.1) so that $Ax \leq b$ is feasible. Then a solution of $Ax \leq b$ can be found with the same time and space complexity of one application of a single-source shortest path algorithm. Moreover, if b is integral then the solution found is half-integral.* ∎

As a consequence of Proposition 3.1 we have that a half-integral solution of $Ax \leq b$ with A fulfilling (3.1) can be found in $O(\min\{mn, \sqrt{n}m \log L\})$ time, where L is a positive integer (greater than 1) so that b_i is at least $-L$ ($i = 1, \ldots, m$) (cf. Bellman [1958], Ford [1956], Moore [1956] and Goldberg [1993]; see also the Preliminaries on the shortest path problem).

From the relation between solutions of $Ax \leq b$, potentials on D_A (as described by (3.4) and (3.5)) and distances between vertices of D_A it follows that each solution x^* of $Ax \leq b$ satisfies:

$$\begin{aligned} x_j^* &\leq (1/2)\max\{\pi(j^+) - \pi(j^-) \mid \pi \text{ potential on } D_A\} \\ &\leq \text{distance}_{D_A}(j^-, j^+)/2 \quad (j = 1, \ldots, n), \end{aligned} \tag{3.6}$$

where $\text{distance}_{D_A}(u, v)$ denotes the distance from u to v in D_A; if v is not reachable from u in D_A then $\text{distance}_{D_A}(u, v) = \infty$. Analogously,

$$\begin{aligned} x_j^* &\geq -(1/2)\max\{\pi(j^-) - \pi(j^+) \mid \pi \text{ potential on } D_A\} \\ &\geq -\text{distance}_{D_A}(j^+, j^-)/2 \quad (j = 1, \ldots, n). \end{aligned} \tag{3.7}$$

The following proposition summarizes this discussion.

Proposition 3.2 *Let b be an m-vector, let A be a matrix satisfying (3.1) and let x^* be a solution of $Ax \leq b$. Then x^* satisfies*

$$-\mathit{distance}_{D_A}(j^+, j^-)/2 \;\leq\; x_j^* \;\leq\; \mathit{distance}_{D_A}(j^-, j^+)/2 \quad (j = 1, \ldots, n).$$

∎

Remark 3.2. Mata [1984] considered the related problem of finding a solution of a system $Bx \leq c$ involving only inequalities of the form $x_j - x_k \leq d$ ($d < 0$). Mata gave a linear-time algorithm for this problem based on the topological ordering algorithm. (One can see that for systems $Bx \leq c$ of this form either the graph D_B is acyclic or the system is infeasible. Hence, in this case, $O(T_{ss}(m,n)) = O(m+n)$.) Systems of inequalities with this form occur in several computer aided design systems for integrated circuit design (cf. the references in Mata [1984]).

Megiddo [1983] showed that any linear system with n variables and m inequalities so that each inequality has at most two nonzero coefficients can be solved in $O(mn^3 \log n)$ time. Recently, Cohen & Megiddo [1994] improved this bound to $O(mn^2(\log m + \log^2 n))$ time and more recently Hochbaum & Naor [1994] presented a new algorithm that runs in $O(mn^2 \log m)$ time.

3.4. An algorithm for finding a $\{0,1\}$-solution of $Cy \leq d$

We now consider the problem of find a $\{0,1\}$-solution of a system $Cy \leq d$ satisfying (3.3).

In order to find a $\{0,1\}$-solution y^* of $Cy \leq d$, if there exists any, we proceed as follows. Let D_C be the graph associated to $Cy \leq d$. We may assume that D_C has no isolated vertices (this would correspond to a zero column of C). For $i = 1, \ldots, n$ (successively) we set the value of y_i^* according to one of the following rules:

- (R.1) D_C has an $i^+ - i^-$-path and no $i^- - i^+$-path. Then we set y_i^* to 1;
- (R.2) D_C has an $i^- - i^+$-path and no $i^+ - i^-$-path. Then we set y_i^* to 0;
- (R.3) D_C has no $i^+ - i^-$-path and no $i^- - i^+$-path. Then we (arbitrarily) set y_i^* either to 1 and add the arc (i^+, i^-) of length -1 to D_C, or to 0 and add the arc (i^-, i^+) of length 1 to D_C (we also denote the modified graph by D_C);
- (R.4) Finally, if i^+ and i^- belong to the same strong component of D_C then the algorithm stops and declares that $Cy \leq d$ has no integral solution.

The algorithm returns y^*.

We first prove the correctness of the algorithm above and after that we

discuss how it can be realized in linear time.

Correctness of the algorithm

The correctness of the algorithm is based on the following three lemmas.

Lemma 3.1 *The arcs added to D_C by the algorithm do not change its strong components.*

Proof. Trivial. ∎

From Lemma 3.1 it follows that the value of each variable y_i is fixed at the very beginning of the algorithm by the reachability between vertices of D_C, up to the arbitrary choice in rule $(R.3)$.

Lemma 3.2 *If there exists no $j \in \{1, \ldots, n\}$ such that j^+ and j^- belong to the same strong component of D_C then the vector y^* returned by the algorithm is a $\{0,1\}$-solution of $Cy \leq d$.*

Proof. Let $j, k \in \{1, \ldots, n\}$ be so that $j < k$ and let D_k denote the graph D_C as it is just before the value of y_k is set. Suppose $y_j^* = 1$ (the case $y_j^* = 0$ is analogous). So by rule $(R.1)$ and $(R.3)$, D_k has a $j^+ - j^-$-path P_{j^+,j^-}. We consider four cases:

Case 1. $y_j - y_k \leq 0$ is an inequality of $Cy \leq d$. Thus (k^+, j^+) and (j^-, k^-) are arcs of D_C and by rule $(R.1)$ and $(R.3)$. So, D_k has a $k^+ - k^-$-path formed by the arcs (k^+, j^+), (j^-, k^-) and the path P_{j^+,j^-}, implying that $y_k^* = 1$ (by rule $(R.1)$).

Case 2. $y_j + y_k \leq 1$ is an inequality of $Cy \leq d$. Thus, (j^-, k^+) and (k^-, j^+) are arcs of D_C. So, D_k has a $k^- - k^+$-path formed by the arcs (k^-, j^+), (j^-, k^+) and the path P_{j^+,j^-}, implying that $y_k^* = 0$ (by rule $(R.2)$).

Case 3. $y_k - y_j \leq 0$ is an inequality of $Cy \leq d$. Trivial, since $y_j^* = 1$.

Case 4. $-y_j - y_k \leq -1$ is an inequality of $Cy \leq d$. Trivial, since $y_j^* = 1$. ∎

Lemma 3.3 *If there exists a $j \in \{1, \ldots, n\}$ so that j^+ and j^- belong to the same strong component of D_C then $Cy \leq d$ has no integral solution (in particular, no $\{0,1\}$-solution).*

Proof. Let P_{j^+,j^-} and P_{j^-,j^+} be a $j^+ - j^-$-path and a $j^- - j^+$-path in D_C, respectively. Since adding the arcs in rule $(R.3)$ does not change strong components, we may assume that P_{j^+,j^-} and P_{j^-,j^+} are in the original graph D_C. From the definition of D_C it follows that

$$\text{length}_{D_C}(P_{j^+,j^-}) = \text{distance}_{D_C}(j^+, j^-) = -1 \quad \text{and}$$
$$\text{length}_{D_C}(P_{j^-,j^+}) = \text{distance}_{D_C}(j^-, j^+) = 1.$$

By Proposition 3.2 we have that any solution y of $Cy \leq d$ must satisfy

$$1/2 = -\text{distance}_{D_C}(j^+, j^-)/2 \;\leq\; y_j \;\leq\; \text{distance}_{D_C}(j^-, j^+)/2 = 1/2.$$

∎

Lemmas 3.2 and 3.3 prove the correctness of the algorithm: either $Cy \leq d$ has a $\{0,1\}$-solution or there exists a $j \in \{1,\ldots,n\}$ such that j^+ and j^- are in the same strong component of D_C.

Complexity of the algorithm

The linear-time realization of the algorithm is as follows:

Step 1. We first compute the strong components $S_1,\ldots,S_t$ of D_C.

Step 2. Let $H = (V, A)$ be the acyclic digraph so that $V = \{S_1,\ldots,S_t\}$ and there is an arc from S_i to S_j if and only if D_C has an arc from a vertex of S_i to a vertex of S_j. By applying a topological ordering algorithm to H and relabeling the strong components we may assume that $(S_i, S_j) \in A$ implies $i < j$.

Step 3. For $i = 1,\ldots,n$ we do as follows:

if $i^+ \in S_j$ and $i^- \in S_k$, $j \neq k$: if $j < k$ then we set y_i^* to 1; if $k < j$ then we set y_i^* to 0;

otherwise (i^+ and i^- belong to the same strong component of D_C) the algorithm stops and returns an $i^- - i^+$-path and an $i^+ - i^-$-path, (by Lemma 3.3 these paths certify the nonexistence of an integral solution of $Cy \leq d$).

The algorithm returns y^*.

One easily verifies that the solution y^* returned by the algorithm, if there exists any, fulfills the conditions of rules $(R.1), (R.2)$ and $(R.3)$. Step 1 can be implemented to run in $O(m+n)$ time (cf. Tarjan [1972]). The topological ordering algorithm runs in $O(m+n)$ time (see, e.g., Tarjan [1983] or Ahuja, Magnanti & Orlin [1993]). Finally, in Step 3, deciding the value of y_i^* takes $O(1)$ time ($i = 1,\ldots,n$) and, if this is the case, an $i^- - i^+$-path and an $i^+ - i^-$-path can be found in $O(m+n)$ time by using breadth-first search. Thus, we have established the following result.

Theorem 3.1 *Let $Cy \leq d$ be a system of inequalities each of one of the following forms:*

$$y_j + y_k \leq 1; \quad y_j - y_k \leq 0; \quad y_k - y_j \leq 0; \quad -y_j - y_k \leq -1.$$

Then the system $Cy \leq d$ has a $\{0,1\}$-solution if and only if there exists no $j \in \{1,\ldots,n\}$ such that j^+ and j^- belong to the same strong component of D_C. Moreover a $\{0,1\}$-solution of $Cy \leq d$ can be found in linear time. ∎

3.5. Rounding a half-integral solution

Let x^* be a half-integral solution of $Ax \leq b$. We construct a system $Cy \leq d$ having one variable y_i for each variable x_i of $Ax \leq b$ $(i = 1, \ldots, n)$. Then we prove that $Cy \leq d$ has a $\{0,1\}$-solution if and only if $Ax \leq b$ has an integral solution. Moreover, if $Ax \leq b$ has an integral solution we show how using a $\{0,1\}$-solution y^* of $Cy \leq d$ one can round x^* into an integral solution of $Ax \leq b$; namely x_i^* should be rounded up if $y_i^* = 1$ and x_i^* should be rounded down if $y_i^* = 0$ $(i = 1, \ldots, n)$.

The system $Cy \leq d$ will merely have from each inequality $a_{i,j}x_j + a_{i,k}x_k \leq b_i$ of $Ax \leq b$, satisfied with equality by nonintegral components x_j^* and x_k^* of x^*, an inequality $a_{i,j}y_j + a_{i,k}y_k \leq d_i$, where the value of d_i depends on whether $(a_{i,j}, a_{i,k}) = (1,1), (1,-1)$ or $(-1,-1)$, as follows:

$$\begin{array}{rcccrcr} x_j^* + x_k^* & = & b_i & \longrightarrow & y_j + y_k & \leq & 1 \\ x_j^* - x_k^* & = & b_i & \longrightarrow & y_j - y_k & \leq & 0 \\ -x_j^* - x_k^* & = & b_i & \longrightarrow & -y_j - y_k & \leq & -1 \end{array} .$$

Constructing the system $Cy \leq d$ takes linear time.

Let $y^* \in \{0,1\}^n$ be such that

$$x_j^* \in \mathbb{Z} \text{ implies } y_j^* = 0 \ (j = 1, \ldots, n). \tag{3.8}$$

One easily verifies that y^* is a solution of $Cy \leq d$ if and only if $\lfloor x^* \rfloor + y^*$ is an integral solution of $Ax \leq b$. (A j so that $x_j^* \in \mathbb{Z}$ corresponds to a zero column of C.) By Theorem 3.1 a $\{0,1\}$-solution y^* of $Cy \leq d$ satisfying (3.8) can be found in linear time. We prove that if $Cy \leq d$ has no $\{0,1\}$-solution then $Ax \leq b$ has no integral solution.

Lemma 3.4 *If there exists a $j \in \{1, \ldots, n\}$ so that j^+ and j^- belong to the same strong component of D_C then*

$$\textit{distance}_{D_A}(j^-, j^+) = -\textit{distance}_{D_A}(j^+, j^-) \quad \textit{and it is odd.}$$

Proof. Let P_{j^+,j^-} and P_{j^-,j^+} be a path from j^+ to j^- and a path from j^- to j^+ in D_C, respectively. P_{j^+,j^-} and P_{j^-,j^+} are also paths in D_A. Let π^* be the potential on D_A defined from x^* as in (3.4). From the definition of $Cy \leq d$ it follows that $\text{length}_{D_A}(u,v) = \pi^*(v) - \pi^*(u)$ for each arc (u,v) in P_{j^+,j^-} and P_{j^-,j^+}. Hence,

$$2x_j^* = \pi^*(j^+) - \pi^*(j^-) \leq \text{distance}_{D_A}(j^-, j^+) \leq \text{length}_{D_A}(P_{j^-,j^+}) = 2x_j^*.$$

Analogously, $\text{distance}_{D_A}(j^+, j^-) = -2x_j^*$. ∎

The following lemma follows immediately from Proposition 3.2.

Lemma 3.5 *If there exists a $j \in \{1, \ldots, n\}$ such that*

$$distance_{D_A}(j^-, j^+) = -distance_{D_A}(j^+, j^-) \quad \text{and it is odd} \tag{3.9}$$

then $Ax \leq b$ has no integral solution. ∎

By Theorem 3.1 if $Cy \leq d$ has no integral solution then there exists a $j \in \{1, \ldots, n\}$ so that j^+ and j^- belong to the same strong component of D_C. By Lemma 3.4 this j satisfies (3.9), implying that $Ax \leq b$ has no integral solution. The following theorems summarize our discussion.

Theorem 3.2 *Let b be an integral m-vector, let A be a matrix satisfying (3.1), and let D_A be the graph associated with the system $Ax \leq b$. Then $Ax \leq b$ has an integral solution if and only if:*

(i) D_A has no negative cycle (equivalently, $Ax \leq b$ is feasible); and
(ii) there exists no $j \in \{1, \ldots, n\}$ such that

$$distance(j^-, j^+) = -distance(j^+, j^-) \quad \text{and it is odd.}$$ ∎

Theorem 3.3 *Let b be an integral m-vector and let A be a matrix satisfying (3.1). If $Ax \leq b$ has an integral solution then every half-integral solution of $Ax \leq b$ can be rounded to an integral solution. Moreover, an integral solution can be found with the same time and space complexity of one application of a single-source shortest path algorithm.* ∎

In one of the steps of Schrijver's algorithm for the disjoint homotopic paths problem (2.1) a system $Ax \leq b$ must be solved in integers, where b is an integral m-vector and A is a matrix fulfilling (3.1) (cf. Schriver [1991]; see also Section 2.3 of this thesis). Moreover, the vector b is so that $b_i \geq -1$ $(i = 1, \ldots, m)$. Hence, it follows from Theorem 3.3 that an integral solution of this system $Ax \leq b$ can be found in $O(\sqrt{n}m) = O(n^{2.5})$ (cf. Goldberg [1993]; see also Section 2.3 of this thesis). Schrijver's original method (using Fourier-Motzkin elimination of variables) to find an integral solution of $Ax \leq b$ runs in $O(n^3)$ time.

3.6. Further notes: 2-satisfiability and bidirected graphs

In the present section we first discuss the equivalence of Theorem 3.1 and results of Even, Itai & Shamir [1976] and Schrijver [1978] on 2-satisfiability. Next, we observe the equivalence between Theorem 3.2 and a theorem due to Hurkens [1989].

2-satisfiability

By the "moreover"-part of Theorem 3.1 and the equivalence between the problem of finding a $\{0, 1\}$-solution of $Cy \leq d$ and 2-satisfiability (see Section 1.4)

we have that 2-satisfiability can also be solved in linear time—a well-known result due to Even, Itai & Shamir [1976].

Schrijver [1978] has shown that a collection of two-literal clauses on U is satisfiable if and only if F contains no "doubly-odd" formula. A *doubly-odd formula* is a collection of clauses of the form

$$(u \vee l_1), (l_2 \vee l_3), \ldots, (l_p \vee u), (\neg u \vee l_{p+1}), (l_{p+2} \vee l_{p+3}), \ldots, (l_q \vee \neg u)$$

satisfying:

(*i*) $u \in U$;

(*ii*) $l_1, \ldots, l_q$ are literals such that $l_{2k-1} \equiv \neg l_{2k} \quad (k = 1, \ldots, q/2)$.

Schrijver's characterization can be seen to be equivalent to the first part of Theorem 3.1. In particular, from a doubly-odd formula of F one can construct (in a very natural way) a cycle in D containing j^+ and j^- for some $j \in \{1, \ldots, n\}$ and vice-versa.

Bidirected graphs and integral solutions of $Ax \leq b$

Let $A = (a_{i,j})$ be an integral $m \times n$-matrix satisfying $\sum_{j=1}^{n} |a_{i,j}| \leq 2$ for all $i = 1, \ldots, m$ and let b be an integral m-vector. Schrijver [1991] characterized in terms of "bidirected graphs" the existence of an integral solution of a system $Ax \leq b$. (As we are interested in integral solutions we may assume that A is a $\{0, \pm 1\}$-matrix—we may replace an inequality $\pm 2x_i \leq b'$ by an inequality $\pm x_i \leq \lfloor b'/2 \rfloor$.)

A *bidirected graph* B_A associated with $Ax \leq b$ is a graph whose vertices are the column indices and its edges are the row indices of A. If j and j' are the nonzero positions of row i (allowing $j = j'$ for inequalities of the type $\pm x_i \leq b_i$) it gives an edge connecting j and j' with signs associated to its ends depending on whether $(a_{i,j}, a_{i,j'}) = (1,1), (1,-1), (-1,1), (-1,-1)$ as show in Figure 3.3. The length of an edge i is b_i if $j \neq j'$ and it is $2b_i$ if $j = j'$ (i.e. i is a loop).

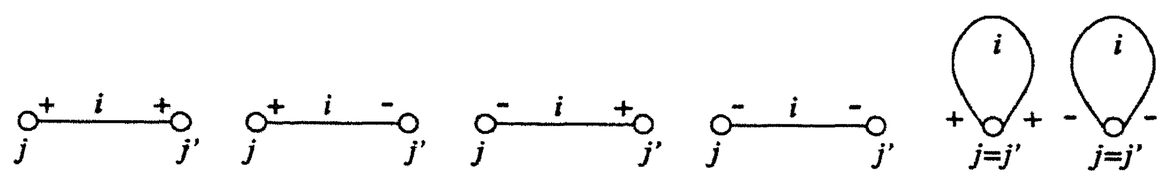


Figure 3.3: Edges of a bidirected graph.

Bidirected graphs were introduced by Jack Edmonds (see Lawler [1976: Ch. 6]).

A *path* in a bidirected graph B_A is a sequence

$$(j_0, i_1, j_1, \ldots, i_t, j_t) \tag{3.10}$$

satisfying:

(*i*) i_h is an edge connecting j_{h-1} and j_h $(h = 1, \ldots, t)$; and

$$(ii)\ a_{i_h,j_h} \cdot a_{i_{h+1},j_h} < 0\ (h = 1, \ldots, t-1).$$

The length of a path is the sum of its edge-lengths. A path (3.10) (with $t \geq 1$) is called a *cycle* if $j_0 = j_t$ and $a_{i_1,j_0} \cdot a_{i_t,j_0} < 0$. A cycle (3.10) is called *doubly-odd* if there exists an s with $0 < s < t$ so that

$$\begin{array}{l} (i)\ j_0 = j_s = j_t \text{ and } a_{i_1,j_0} \cdot a_{i_s,j_s} > 0; \text{ and} \\ (ii)\ \sum_{h=1}^{s} l(i_h) \text{ and } \sum_{h=s+1}^{t} l(i_h) \text{ are odd numbers.} \end{array} \tag{3.11}$$

We shall say that a path $P' = (j'_0, i'_1, j'_1, \ldots, i'_{t'}, j'_{t'})$ *connects* to a path $P'' = (j''_0, i''_1, j''_1, \ldots, i''_{t''}, j''_{t''})$ if $j'_{t'} = j''_0$ and $(a_{i'_{t'},j'_{t'}} \cdot a_{i''_1 j''_0} < 0$ or $t't'' = 0)$. So, when we write $P' \cdot P''$ (the concatenation of P' to P'') we are implicitly assuming that P' connects to P''.

Hurkens [1989], sharpening Schrijver's characterization, has shown the following theorem.

Theorem 3.4 (Hurkens [1989]) *Let b be an integral m-vector, let A be a matrix satisfying (3.1), and let B_A be the bidirected graph associated with the system $Ax \leq b$. Then $Ax \leq b$ has an integral solution if and only if*

(i) each "simple" or "semisimple cycle" in B_A has nonnegative length; and

(ii) each doubly-odd "Korach cycle" in B_A has positive length. ∎

A cycle is *simple* if all its internal vertices are distinct. A cycle C is a *Korach cycle* if there exist paths $P_1, \ldots, P_k, Q_1, \ldots, Q_{2k}$ $(k \geq 1)$ so that

$$\begin{aligned} C \ = \ & P_1 \cdot Q_1 \cdot P_2 \cdot \ \cdots \ \cdot Q_{k-1} \cdot P_k \cdot Q_k \cdot \\ & \cdot P_1^{-1} \cdot Q_{k+1} \cdot P_2^{-1} \cdot Q_{k+2} \cdot \ \cdots \ \cdot Q_{2k-1} \cdot P_k^{-1} \cdot Q_{2k}, \end{aligned} \tag{3.12}$$

where

$$\begin{array}{l} (i) \text{ all vertices in } P_1, \ldots, P_k \text{ and all internal vertices in} \\ \quad Q_1, \ldots, Q_{2k} \text{ are distinct;} \\ (ii)\ Q_i \text{ does not connect to } Q_{k+i+1}\ (i = 1, \ldots, k-1), \\ \quad \text{and } Q_{k+i-1} \text{ does not connect to } Q_i\ (i = 1, \ldots, k+1). \end{array} \tag{3.13}$$

A *semisimple cycle* is a Korach cycle for $k = 1$. Figure 3.4 shows examples of Korach cycles.

There is a correspondence between simple and semisimple negative-length cycles in B_A and negative-length cycles in D_A. Moreover, if we suppose

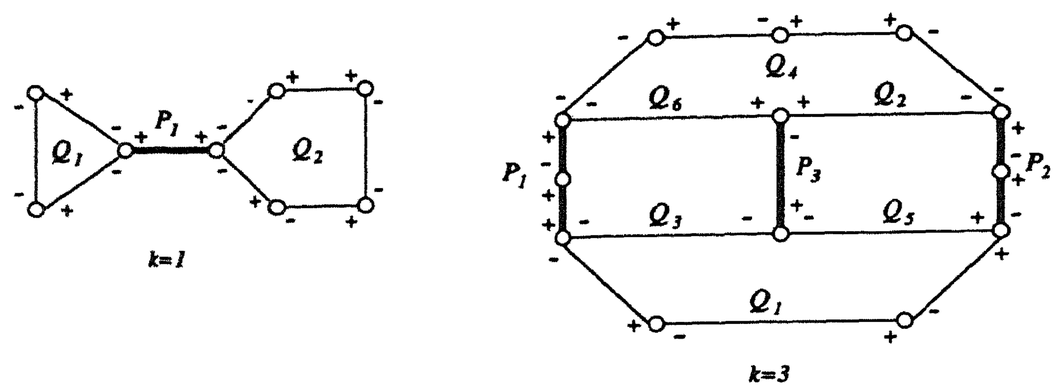


Figure 3.4: Korach cycles.

that B_A has no negative-length cycle (equivalently D_A has no negative-length cycle) then B_A has a zero-length doubly-odd Korach cycle if and only if there exists a $j \in \{1, \ldots, n\}$ so that $\text{distance}_{D_A}(j^+, j^-) = -\text{distance}_{D_A}(j^-, j^+)$ and it is odd. This discussion is the contents of the following theorem.

Theorem 3.5 *Hurkens' theorem (Theorem 3.4) is equivalent to Theorem 3.2.*

Proof. We first describe the notation we shall use along the proof. For each column index j of A there is in B_A a corresponding vertex, also denoted by j, and there are in D_A two corresponding distinct vertices, denoted by j^+ and j^-. We define $\overline{j^+} := j^-$ and $\overline{j^-} := j^+$.

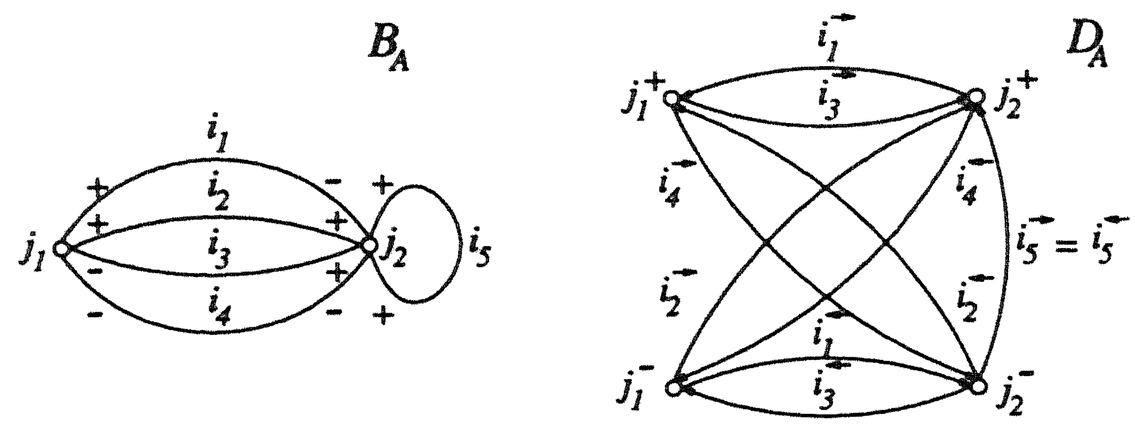
For each row index i of matrix A there is in B_A one corresponding edge, denoted by i, and there are in D_A two corresponding arcs, denoted by $i^{\rightarrow}$ and $i^{\leftarrow}$. (Observe that by the definition of D_A: either $i^{\rightarrow}$ and $i^{\leftarrow}$ are distinct arcs corresponding to the same inequality of $Ax \leq b$; or $i^{\rightarrow} = i^{\leftarrow} = (l^+, l^-)$; or $i^{\rightarrow} = i^{\leftarrow} = (l^-, l^+)$ for some $l \in \{1, \ldots, n\}$.) For each arc $a = (u, v)$ of D_A we define $\overline{a} := (\overline{v}, \overline{u})$. (Note that by the definition of D_A, $\overline{a}$ is also an arc of D_A. Moreover, observe that $\overline{i^{\rightarrow}} = i^{\leftarrow}$ and $\overline{i^{\leftarrow}} = i^{\rightarrow}$.)

If $P^{\rightarrow} = (j_0^{\rightarrow}, i_1^{\rightarrow}, j_1^{\rightarrow}, \ldots, i_t^{\rightarrow}, j_t^{\rightarrow})$ is a path in D_A then

$$\overline{P^{\rightarrow}} \quad := \quad (\overline{j_t^{\rightarrow}}, \overline{i_t^{\rightarrow}}, \overline{j_{t-1}^{\rightarrow}}, \ldots, \overline{i_1^{\rightarrow}}, \overline{j_0^{\rightarrow}})$$

is a path in D_A (this follows directly from the definition of D_A, see Section 3.2—it is possible that $P^{\rightarrow}$ and $\overline{P^{\rightarrow}}$ are the same path, see Figure 3.5). Note also that $\text{length}_{D_A}(P^{\rightarrow}) = \text{length}_{D_A}(\overline{P^{\rightarrow}})$.

We should also remark that for each path $P = (j_0, i_1, j_1, \ldots, i_t, j_t)$ in B_A there exists in D_A a path (possibly not unique) $P^{\rightarrow} = (j_0^{\rightarrow}, i_1^{\rightarrow}, j_1^{\rightarrow}, \ldots, i_t^{\rightarrow}, j_t^{\rightarrow})$ so that $j_k^{\rightarrow} \in \{j_k^-, j_k^+\}$ $(k = 0, \ldots, t)$ and the arc $i_k^{\rightarrow}$ and edge i_k correspond to the same inequality of $Ax \leq b$ $(k = 1, \ldots, t)$. Conversely, for each path $P^{\rightarrow} = (j_0^{\rightarrow}, i_1^{\rightarrow}, j_1^{\rightarrow}, \ldots, i_t^{\rightarrow}, j_t^{\rightarrow})$ there exists a *unique* path $P = (j_0, i_1, j_1, \ldots, i_t, j_t)$ so that $j_k^{\rightarrow} \in \{j_k^-, j_k^+\}$ $(k = 0, \ldots, t)$ and the arc $i_k^{\rightarrow}$ and edge i_k correspond to the same inequality of $Ax \leq b$ $(k = 1, \ldots, t)$.



(a) (b)

Figure 3.5: Figure (a) shows a bidirected graph B_A and Figure (b) shows the corresponding digraph D_A. If $P^{\rightarrow} = (j_1^-, \overleftarrow{i_1}, j_2^-, \overrightarrow{i_5}, j_2^+, \overrightarrow{i_1}, j_1^-)$ and $Q^{\rightarrow} = (j_1^-, \overleftarrow{i_1}, j_2^-, \overleftarrow{i_2}, j_1^+, \overrightarrow{i_3}, j_2^+)$ then $P^{\rightarrow} = \overline{P^{\rightarrow}}$ and $\overline{Q^{\rightarrow}} = (j_2^-, \overleftarrow{i_3}, j_1^-, \overrightarrow{i_2}, j_2^+, \overrightarrow{i_1}, j_1^+)$. Note that associated with the Korach cycle (for $k = 2$) $(j_1, i_1, j_2, i_2, j_1, i_3, j_2, i_4, j_1)$ there are two directed circuits in D_A, namely, $(j_1^+, \overrightarrow{i_3}, j_2^+, \overleftarrow{i_4}, j_1^-, \overleftarrow{i_1}, j_2^-, \overleftarrow{i_2}, j_1^+)$ and $(j_1^+, \overrightarrow{i_4}, j_2^-, \overleftarrow{i_3}, j_1^-, \overrightarrow{i_2}, j_2^+, \overrightarrow{i_1}, j_1^+)$.

I. B_A has a simple or semisimple negative-length cycle $\Leftrightarrow$ D_A has a negative-length cycle.

($\Rightarrow$) As we have already observed, it follows immediately from the definition of D_A that if $(j_0, i_1, j_1, \ldots, i_t, j_t)$ is a cycle in B_A then there exists in D_A a cycle $(\overrightarrow{j_0}, \overrightarrow{i_1}, \overrightarrow{j_1}, \ldots, \overrightarrow{i_t}, \overrightarrow{j_t})$ so that $\overrightarrow{j_k} \in \{j_k^-, j_k^+\}$ $(k = 0, \ldots, t)$. Thus, if B_A has a negative-length cycle (simple or semisimple) then D_A has a negative-length cycle.

($\Leftarrow$) Let $C^{\rightarrow} = (\overrightarrow{j_0}, \overrightarrow{i_1}, \overrightarrow{j_1}, \ldots, \overrightarrow{i_t}, \overrightarrow{j_t})$ be a negative-length circuit in D_A and let $C = (j_0, i_1, j_1, \ldots, i_t, j_t)$ be the cycle in B_A associated with $C^{\rightarrow}$. Suppose we have chosen $C^{\rightarrow}$ so that $|\{i_1, \ldots, i_t\}|$ is as small as possible (equivalently, we are choosing a infeasible subsystem $A'x \leq b'$ of $Ax \leq b$ with as few inequalities as possible).

If at most one vertex of B_A is traversed twice by C then we are done (note that no vertex of C can be traversed more than twice since $C^{\rightarrow}$ is a circuit). So, suppose there exist distinct indices u, v, p, q, $0 \leq u < \min\{v, p\} < \max\{v, p\} < q$ so that $j_u = j_p$ and $j_v = j_q$. Thus, (as $C^{\rightarrow}$ is a circuit) $\{\overrightarrow{j_u}, \overrightarrow{j_p}\} = \{j_u^-, j_u^+\}$ and $\{\overrightarrow{j_v}, \overrightarrow{j_q}\} = \{j_v^-, j_v^+\}$. We need to consider two possibilities.

Case 1. $u < v < p < q$.

We can write

$$C^{\rightarrow} = P_1^{\rightarrow} \cdot Q_1^{\rightarrow} \cdot P_2^{\rightarrow} \cdot Q_2^{\rightarrow},$$

where $P_1^{\rightarrow}$ is a path from $j_u^{\rightarrow}$ to $j^{\rightarrow}j_v$, $Q_1^{\rightarrow}$ is a path from $j_v^{\rightarrow}$ to $j_p^{\rightarrow}$, $P_2^{\rightarrow}$ is a path from $j_p^{\rightarrow}$ to $j_q^{\rightarrow}$, $Q_2^{\rightarrow}$ is a path from $j_q^{\rightarrow}$ to $j_u^{\rightarrow}$.

At least one of the cycles $P_1^{\rightarrow} \cdot \overline{P_2^{\rightarrow}}$ or $Q_1^{\rightarrow} \cdot \overline{Q_2^{\rightarrow}}$ has negative length. This contradicts our minimality assumption. Thus, Case 1 does not occur.

Case 2. $u < p < v < q$.

We can write

$$C^{\rightarrow} = P_1^{\rightarrow} \cdot Q_1^{\rightarrow} \cdot P_2^{\rightarrow} \cdot Q_2^{\rightarrow},$$

where $P_1^{\rightarrow}$ is a path from $j_u^{\rightarrow}$ to $j_p^{\rightarrow}$, $Q_1^{\rightarrow}$ is a path from $j_p^{\rightarrow}$ to $j_v^{\rightarrow}$, $P_2^{\rightarrow}$ is a path from $j_v^{\rightarrow}$ to $j_q^{\rightarrow}$, $Q_2^{\rightarrow}$ is a path from $j_q^{\rightarrow}$ to $j_u^{\rightarrow}$.

By our minimality assumption it follows that $C^{\rightarrow} = P_1^{\rightarrow} \cdot Q_1^{\rightarrow} \cdot \overline{P_1^{\rightarrow}} \cdot Q_2^{\rightarrow}$ or $C^{\rightarrow} = \overline{P_2^{\rightarrow}} \cdot Q_1^{\rightarrow} \cdot P_2^{\rightarrow} \cdot Q_2^{\rightarrow}$ (as at least one of these two cycles have negative length). This completes the analysis of Case 2.

Hence, if we suppose that C traverses distinct vertices j_u and j_v more than once then by Case 1 we conclude that C traverses j_u and j_v exactly twice, once when going from j_u to j_v and once when going from j_v to j_u and by Case 2 we have that the paths in C from j_u to j_v and from j_v to j_u are the reverse to each other. Therefore, C is a semisimple cycle.

II. B_A has a zero-length doubly-odd Korach cycle $\Leftrightarrow$ there exists a $j \in \{1, \ldots, n\}$ so that $\text{distance}_{D_A}(j^+, j^-) = -\text{distance}_{D_A}(j^-, j^+)$ and it is odd.

Recall that we are assuming that B_A (and therefore D_A) has no negative cycle. This implies that if $C^{\rightarrow}$ is a zero-length cycle in D_A then any subpath of $C^{\rightarrow}$ is a shortest path in D_A.

($\Rightarrow$) Let $C = (j_0, i_1, j_1, \ldots, i_t, j_t)$ be a zero-length doubly-odd Korach cycle in B_A and let s, $0 < s < t$, be an index fulfilling (3.11). Let $C^{\rightarrow} = (j_0^{\rightarrow}, i_1^{\rightarrow}, j_1^{\rightarrow}, \ldots, i_t^{\rightarrow}, j_t^{\rightarrow})$ be a cycle in D_A associated with C. By (3.11)(i) it follows that $\{j_0^{\rightarrow}, j_s^{\rightarrow}\} = \{j_0^-, j_0^+\}$. Let $R^{\rightarrow}$ and $S^{\rightarrow}$ be the path in $C^{\rightarrow}$ from $j_0^{\rightarrow}$ to $j_s^{\rightarrow}$ and the path in $C^{\rightarrow}$ from $j_s^{\rightarrow}$ to $j_t^{\rightarrow}$, respectively. Since C is a zero-length cycle, then by (3.11)(ii) we have that

$$\text{length}_{D_A}(R^{\rightarrow}) = -\text{length}_{D_A}(S^{\rightarrow}) \text{ and it is odd}$$

(otherwise either $R^{\rightarrow} \cdot \overline{R^{\rightarrow}}$ or $S^{\rightarrow} \cdot \overline{S^{\rightarrow}}$ would be a negative-length cycle in D_A). Moreover, $R^{\rightarrow}$ and $S^{\rightarrow}$ are shortest paths in D_A. Therefore, $j := j_0 \in \{1, \ldots, n\}$ is so that $\text{distance}_{D_A}(j^+, j^-) = -\text{distance}_{D_A}(j^-, j^+)$ and it is odd.

($\Leftarrow$) Let $j \in \{1, \ldots, n\}$ so that $\text{distance}_{D_A}(j^+, j^-) = -\text{distance}_{D_A}(j^-, j^+)$ and it is odd. Let $R^{\rightarrow} = (\vec{j}_0, \vec{i}_1, \vec{j}_1, \ldots, \vec{i}_s, \vec{j}_s)$ be a shortest simple path from j^- to j^+ in D_A and let $S^{\rightarrow} = (\vec{j}_s, \vec{i}_{s+1}, \vec{j}_{s+1}, \ldots, \vec{i}_t, \vec{j}_t)$ be a shortest simple path from j^+ to j^- in D_A. Thus $\vec{C} = R^{\rightarrow} \cdot S^{\rightarrow}$ is a zero-length cycle in D_A. Let

$C := (j_0, i_1, j_1, \ldots, i_t, j_t)$ be the cycle in B_A associated with $C^{\rightarrow}$. By definition it follows that C is a zero-length doubly-odd cycle (the index s fulfills (3.11)). We show that C is Korach cycle.

We may assume that we have chosen j, $R^{\rightarrow}$ and $S^{\rightarrow}$ in such a way that:

(*i*) $|\{i_1, \ldots, i_t\}|$ is as small as possible;

(*ii*) the number of arcs in $C^{\rightarrow}$ is as small as possible.

In this order ((*ii*) should hold under condition (*i*)).

Claim 3.1 *Suppose that $R^{\rightarrow}$ and $S^{\rightarrow}$ have an internal vertex in common. Then there exists no $l \in \{1, \ldots, n\} \setminus \{j\}$ so that $l^{\rightarrow}$ is a vertex of $R^{\rightarrow}$ and $\overline{l^{\rightarrow}}$ is a vertex of $S^{\rightarrow}$.*

Proof. Let us suppose by contradiction that there exist a vertex $u^{\rightarrow}$ of D_A traversed by both $S^{\rightarrow}$ and $R^{\rightarrow}$ and an $l \in \{1, \ldots, n\}$ satisfying the conditions of the claim. We consider here the case when $R^{\rightarrow}$ traverses first $u^{\rightarrow}$ and then $l^{\rightarrow}$ and $S^{\rightarrow}$ traverses first $u^{\rightarrow}$ and then $\overline{l^{\rightarrow}}$; the other case ($R^{\rightarrow}$ traverses first $l^{\rightarrow}$ and then $u^{\rightarrow}$) can be handled in a similar manner.

Let $R_1^{\rightarrow}, R_2^{\rightarrow}, R_3^{\rightarrow}, S_1^{\rightarrow}, S_2^{\rightarrow}$ and $S_3^{\rightarrow}$ be paths so that

$$\begin{aligned} R^{\rightarrow} &= R_1^{\rightarrow} \cdot R_2^{\rightarrow} \cdot R_3^{\rightarrow} \quad \text{and} \\ S^{\rightarrow} &= S_1^{\rightarrow} \cdot S_2^{\rightarrow} \cdot S_3^{\rightarrow}, \end{aligned}$$

where $R_1^{\rightarrow}$ is a path from j^- to $u^{\rightarrow}$, $R_2^{\rightarrow}$ is a path from $u^{\rightarrow}$ to $l^{\rightarrow}$, $R_3^{\rightarrow}$ is a path from $l^{\rightarrow}$ to j^+, $S_1^{\rightarrow}$ is a path from j^+ to $u^{\rightarrow}$, $S_2^{\rightarrow}$ is a path from $u^{\rightarrow}$ to $\overline{l^{\rightarrow}}$ and $S_3^{\rightarrow}$ is a path from $\overline{l^{\rightarrow}}$ to j^-.

Since $C^{\rightarrow}$ is a zero-length cycle, then $C_1^{\rightarrow} := \overline{S_1^{\rightarrow}} \cdot R_1^{\rightarrow} \cdot R_2^{\rightarrow} \cdot \overline{S_2^{\rightarrow}}$ and $C_2^{\rightarrow} := R_3^{\rightarrow} \cdot \overline{S_3^{\rightarrow}}$ are both zero-length cycles. We show that

$$\text{length}_{D_A}(\overline{S_1^{\rightarrow}}) + \text{length}_{D_A}(R_1^{\rightarrow})$$

is odd. This implies that the cycle $C_1^{\rightarrow}$ contradicts the minimality assumption since $\overline{S_1^{\rightarrow}} \cdot R_1^{\rightarrow}$ is a shortest path from $\overline{u^{\rightarrow}}$ to $u^{\rightarrow}$, $R_2^{\rightarrow} \cdot \overline{S_2^{\rightarrow}}$ is a shortest path from $u^{\rightarrow}$ to $\overline{u^{\rightarrow}}$ and $C_1^{\rightarrow}$ has fewer arcs than $C^{\rightarrow}$.

Indeed, $S_1^{\rightarrow} \cdot R_2^{\rightarrow} \cdot R_3^{\rightarrow}$ and $R_1^{\rightarrow} \cdot S_2^{\rightarrow} \cdot S_3^{\rightarrow}$ are zero-length cycles. Hence,

$$\begin{aligned} &\text{length}_{D_A}(\overline{S_1^{\rightarrow}}) + \text{length}_{D_A}(R_1^{\rightarrow}) = \\ &\quad -(\text{length}_{D_A}(R_2^{\rightarrow}) + \text{length}_{D_A}(R_3^{\rightarrow})) + \text{length}_{D_A}(R_1^{\rightarrow}) \end{aligned}$$

which is odd since $\text{length}_{D_A}(R^{\rightarrow})$ is odd. ∎

We may assume that $R^{\rightarrow}$ and $S^{\rightarrow}$ are internally vertex disjoint. Indeed, suppose to the contrary. From Claim 3.1 it follows that $R^{\rightarrow}$ and $\overline{S^{\rightarrow}}$ are internally vertex disjoint. Hence, we may replace $S^{\rightarrow}$ by $\overline{S^{\rightarrow}}$.

Without loss of generality we may also assume that path $S^{\rightarrow}$ does not traverse both l^- and l^+, for each $l \in \{1,\ldots,n\}\setminus\{j\}$. This assumption together with the minimality assumption imply that there exist paths $P_1^{\rightarrow}$ and $Q^{\rightarrow}$ in D_A so that

$$R^{\rightarrow} = P_1^{\rightarrow} \cdot Q^{\rightarrow} \cdot \overline{P_1^{\rightarrow}} ,$$

where $Q^{\rightarrow}$ satisfies $|V(Q^{\rightarrow}) \cap \{l^-, l^+\}| \leq 1$ for each $l \in \{1,\ldots,n\} \setminus \{j\}$.

Let $Q_1^{\rightarrow}, P_2^{\rightarrow}, Q_2^{\rightarrow}, \ldots, P_k^{\rightarrow}, Q_k^{\rightarrow}$ be subpaths of $R^{\rightarrow}$ so that:
(*i*) $R^{\rightarrow} = P_1^{\rightarrow} \cdot Q_1^{\rightarrow} \cdot P_2^{\rightarrow} \cdot Q_2^{\rightarrow} \cdot \cdots P_k^{\rightarrow} \cdot Q_k^{\rightarrow} \cdot \overline{P_1^{\rightarrow}}$;
(*ii*) for $i = 2,\ldots,k$ it holds that $P_i^{\rightarrow}$ is a maximal subpath (with respect to the number of arcs) of $R^{\rightarrow}$ so that $\overline{P_i^{\rightarrow}}$ is a subpath of $S^{\rightarrow}$.

Claim 3.2 *Let i,j be indices so that $2 \leq i < j \leq k$. Then $S^{\rightarrow}$ traverses $\overline{P_i^{\rightarrow}}$ before $\overline{P_j^{\rightarrow}}$.*

Proof. Without loss of generality we assume that $(i,j) = (2,3)$. Let us suppose by contradiction that there exist paths $\vec{R}_1, \vec{R}_2, \vec{R}_3, \vec{S}_1, \vec{S}_2$ and $\vec{S}_3$ so that

$$\begin{aligned} R^{\rightarrow} &= \vec{R}_1 \cdot P_2^{\rightarrow} \cdot \vec{R}_2 \cdot P_3^{\rightarrow} \cdot \vec{R}_3 \quad \text{and} \\ S^{\rightarrow} &= \vec{S}_1 \cdot \overline{P_3^{\rightarrow}} \cdot \vec{S}_2 \cdot \overline{P_2^{\rightarrow}} \cdot \vec{S}_3 . \end{aligned}$$

By the minimality assumption it follows that $S_2^{\rightarrow} = \overline{R_2^{\rightarrow}}$. This contradicts the definition of $P_2^{\rightarrow}$ and $P_3^{\rightarrow}$. ∎

From Claim 3.2 it follows that there exist paths $Q_{k+1}^{\rightarrow}, Q_{k+2}^{\rightarrow}, \ldots, Q_{2k}^{\rightarrow}$ so that

$$S^{\rightarrow} = \overline{P_2^{\rightarrow}} \cdot Q_{k+1}^{\rightarrow} \cdot \overline{P_3^{\rightarrow}} \cdot Q_{k+2}^{\rightarrow} \cdot \cdots \cdot \overline{P_k^{\rightarrow}} \cdot Q_{2k}^{\rightarrow}.$$

(Note that each $Q_i^{\rightarrow}$ has at least one arc.) Hence,

$$\begin{aligned} C^{\rightarrow} = {} & P_1^{\rightarrow} \cdot Q_1^{\rightarrow} \cdot P_2^{\rightarrow} \cdot \cdots \cdot Q_{k-1}^{\rightarrow} \cdot P_k^{\rightarrow} \cdot Q_k^{\rightarrow} \cdot \\ & \cdot \overline{P_1^{\rightarrow}} \cdot Q_{k+1}^{\rightarrow} \cdot \overline{P_2^{\rightarrow}} \cdot Q_{k+2}^{\rightarrow} \cdot \cdots \cdot Q_{2k-1}^{\rightarrow} \cdot \overline{P_k^{\rightarrow}} \cdot Q_{2k}^{\rightarrow}. \end{aligned}$$

Therefore, cycle C has the form displayed in (3.12).

Condition (3.13)(*i*) follows from the fact that $R^{\rightarrow}$ and $S^{\rightarrow}$ are internally vertex-disjoint paths.

Note now that the endpoint of $Q_i^{\rightarrow}$ and the begin point of $Q_{k+i+1}^{\rightarrow}$ do not coincide ($i = 1,\ldots,k-1$) and the endpoint of $Q_{k+i-1}^{\rightarrow}$ and the begin point of $Q_i^{\rightarrow}$ do not coincide either ($i = 1,\ldots,k+1$). From this and the definition of the graph D_A one can derive (3.13)(*ii*). Therefore, C is a Korach cycle. This completes the proof of Theorem 3.5. ∎

Chapter 4

Optimization of integral potentials

4.1. Introduction

Let b be an integral m-vector and let A be matrix satisfying

$A = (a_{i,j})$ is an $m \times n - \{0, \pm 1\}$-matrix with at most two nonzero entries per row. (4.1)

In this chapter we consider the problem:

$$\max\{c^T x \mid Ax \leq b,\ x \text{ integral}\}, \tag{4.2}$$

This optimization problem is $\mathcal{NP}$-hard in the strong sense. (One can easily reduce the *maximum independent set problem* (see, e.g., Garey & Johnson [1979]) to problem (4.2) restricted to instances where all inequalities in $Ax \leq b$ are of the form $x_i + x_j \leq 1$ and $c^T = (1, \ldots, 1)$.)

However, as we observe here, problem (4.2) can be reduced to n applications of a single-source shortest path algorithm (or to one application of an all-pairs shortest path algorithm) provided that $c \geq 0$ and

that matrix A has no more than one positive element in any row[1]. (4.3)

(Observe that deleting any of the previous conditions problem (4.2) is $\mathcal{NP}$-hard.)

Let P be a polyhedron in $\mathbb{R}^n$. We say x^* is a *maximal element* of P if $\max\{c^T x \mid x \in P\}$ is achieved by x^* for each vector $c \geq 0$ for which the maximum is finite. (Likewise, x^* is *minimal element* of P if $\min\{c^T x \mid x \in P\}$ is achieved by x^* for each vector $c \geq 0$ for which the minimum is finite.)

Polyhedral sets having a maximal (minimal) element have been characterized, in terms of its inequalities system, by Cottle & Veinott [1972].

[1] A matrix with no more than one positive element in any column is called a *pre-Leontief matrix*. Thus, A^T is pre-Leontief.

Adler & Cosares [1991], combining results of Cottle & Veinott [1972] and Megiddo [1983], have shown that if A is a matrix (not necessarily a $\{0, \pm 1\}$-matrix) with no more than two nonzero elements per row satisfying (4.3) then optimum solutions of

$$\max\{c^T x \mid Ax \leq b\} = \min\{b^T y \mid y^T A = c,\ y \geq 0\}$$

can be found in $O(m^3 n \log n)$ time (thus, strongly polynomial time) provided that the optimum is finite. In particular, Adler & Cosares' algorithm implies the strongly polynomial solvability of a class of *generalized flow problems* (see, e.g., Ahuja, Magnanti & Orlin [1993: Ch. 15]). Recently, Cohen & Megiddo [1994] have improved this bound to $O(mn^2(\log m + \log^2 n))$ time and Hochbaum & Naor [1994] presented a $O(mn^2 \log m)$-time algorithm.

Let A be a matrix satisfying (4.1) and (4.3). In Section 4.2 we show that if $Ax \leq b$ is feasible then $\{x \mid Ax \leq b\}$ has a maximal element and it can be found in $O(nT_{ss}(m,n))$ time (or $O(T_{all}(m,n))$ time). From this we derive that $\max\{c^T x \mid Ax \leq b\}$ can be solved in $O(nT_{ss}(m,n))$ time for each vector $c \geq 0$. Next, in Section 4.3, we show that if in addition b is an integral m-vector and $Ax \leq b$ has an integral solution then $\{x \mid Ax \leq b\}$ has an integral maximal element, implying that $\max\{c^T x \mid Ax \leq b\ x \text{ integral}\}$ can be solved in $O(nT_{ss}(m,n))$ time for each vector $c \geq 0$.

We use the following notation. If D_A is the digraph associated to a system $Ax \leq b$ as defined in Section 3.2 then we shall denote the sets $\{j^+ \mid j \in \{1,\ldots,n\}\}$ and $\{j^- \mid j \in \{1,\ldots,n\}\}$ by V^+ and V^-, respectively.

4.2. Finding a maximal element

Let A be a matrix satisfying (4.1) and (4.3). Let us assume that $Ax \leq b$ is feasible (equivalently, D_A has no negative cycle). Let y^* be a solution of $Ax \leq b$ (by Proposition 3.1 such a y^* can be found in $O(T_{ss}(m,n))$ time).

Our candidate for a maximal element of the polyhedron $\{x \mid Ax \leq b\}$ is the vector x^* defined as follows, for $i = 1.\ldots,n$:

$$x_i^* := \begin{cases} \text{distance}_{D_A}(i^-, i^+)/2 & \text{if } i^+ \text{ is reachable from } i^-, \\ y_i^* & \text{otherwise.} \end{cases} \tag{4.4}$$

By Proposition 3.2 it is clear that if x^* is a solution of $Ax \leq b$ then it is a maximal element of $\{x \mid Ax \leq b\}$. It is also clear that x^* can be computed in $O(nT_{ss}(m,n))$ time (or $O(T_{all}(m,n))$ time).

Lemma 4.1 *Let x^* be an n-vector as defined in (4.4) and let π^* be the function on the vertex set of D_A defined as follows:*

$$\pi^*(i^+) := x_i^*, \quad \pi^*(i^-) := -x_i^* \quad (i = 1,\ldots,n).$$

Then π^* is a potential function on D_A.

Proof. Let $j, k \in \{1, \ldots, n\}$, $j \neq k$ so that (j^+, k^-) is an arc of D_A (so, (k^+, j^-) is also an arc of D_A). We verify that $\pi^*(k^-) - \pi^*(j^+) \leq \text{length}_{D_A}(j^+, k^-)$. The verification that π^* is a potential for the other types of arcs of D_A is analogous. We need to consider four possibilities.

Case 1. j^+ is reachable from j^- and k^+ is reachable from k^-. Thus,

$$\begin{aligned} \pi^*(k^-) - \pi^*(j^+) &= -\text{distance}_{D_A}(k^-, k^+)/2 - \text{distance}_{D_A}(j^-, j^+)/2 \\ &\leq \text{length}_{D_A}(j^+, k^-) \quad (\text{as } D_A \text{ has no negative cycle}). \end{aligned}$$

Case 2. j^+ reachable from j^- and k^+ is not reachable from k^-. In this case we have that

$$\begin{aligned} \pi^*(k^-) - \pi^*(j^+) &= -y_k^* - \text{distance}_{D_A}(j^-, j^+)/2 \\ &\leq -y_k^* - y_j^* \quad (\text{by } (3.2)) \\ &\leq \text{length}_{D_A}(j^+, k^-) \quad (\text{as } y^* \text{ is a solution of } Ax \leq b). \end{aligned}$$

Case 3. j^+ is not reachable from j^- and k^+ is reachable from k^-. Thus,

$$\begin{aligned} \pi^*(k^-) - \pi^*(j^+) &= -\text{distance}_{D_A}(k^-, k^+)/2 - y_j^* \\ &\leq -y_k^* - y_j^* \quad (\text{by } (3.2)) \\ &\leq \text{length}_{D_A}(j^+, k^-) \quad (\text{as } y^* \text{ is a solution of } Ax \leq b). \end{aligned}$$

Case 4. j^+ is not reachable from j^- and k^+ is not reachable from k^-. Thus $\pi^*(k^-) - \pi^*(j^+) = -y_k^* - y_j^* \leq \text{length}_{D_A}(j^+, k^-)$, as y^* is a solution of $Ax \leq b$. ∎

From Lemma 4.1 and Proposition 3.2 we have:

Theorem 4.1 *Let b be m-vector and let A be a matrix satisfying (4.1) and (4.3). Then $\{x \mid Ax \leq b\}$ has a maximal element, provided that $Ax \leq b$ is feasible. Moreover a maximal element of $\{x \mid Ax \leq b\}$ can be found in $O(nT_{ss}(m, n))$ time.* ∎

Corollary 4.1 *Let b be m-vector and A be a matrix satisfying (4.1) and (4.3). Then $\max\{c^T x \mid Ax \leq b\}$ can be solved in $O(nT_{ss}(m, n))$ time for each $c \geq 0$.*

Before proving Corollary 4.1 we prove the following Lemma.

Lemma 4.2 *For each $j \in \{1, \ldots, n\}$ one has: j^+ is not reachable from j^- if and only if there exists a $z^* \geq 0$, $z_j^* = 1$ so that $Az^* \leq 0$.*

Proof. From Proposition 3.2 it follows that if $D_{A,0}$ has a path from j^- to j^+ then $z_j^* \leq 0$ for each solution z^* of $Ax \leq 0$ $(j = 1, \ldots, n)$. Conversely, suppose j^+ is not reachable from j^- in $D_{A,0}$. Let V_j be the set of vertices u so that j^+ is reachable from u. From the definition of $D_{A,0}$ and condition (4.3) we have that $V_j \subseteq V^+$ (otherwise $D_{A,0}$ would contain a path from j^- to j^+—since the arcs of $D_{A,0}$ from a vertex in V^- to a vertex of V^+ are of the type (k^-, k^+), $k \in \{1, \ldots, n\}$). We define a function π^* on the vertices of $D_{A,0}$ as follows:

$$\pi^*(u) := \begin{cases} 2 & \text{if } u \in V_j, \\ 0 & \text{otherwise} \end{cases} \qquad (i = 1, \ldots, n).$$

π^* is a potential function on $D_{A,0}$ so that z^* defined from π^* as in (3.5) is a solution of $Ax \leq 0$ satisfying the conditions of the Lemma. ∎

Proof of Corollary 4.1. In $O(T_{ss}(m,n))$ time we can either compute a solution y^* of $Ax \leq b$ or find a negative cycle in D_A (cf. Proposition 3.1). Let us assume that $Ax \leq b$ is feasible, otherwise we are done. Let $C^+ := \{i \in \{1, \ldots, n\} \mid c_i > 0\}$ and let x^* be a maximal element of $\{x \mid Ax \leq b\}$ as defined in (4.4) (by Theorem 4.1, x^* can be computed in $O(nT_{ss}(m,n)))$ time. The maximum is finite if and only if there exists no z^* so that $Az^* \leq 0$, $c^T z > 0$. Hence, by Lemma 4.2, if there exist a $j \in C^+$ so that j^+ is not reachable from j^- then the maximum is infinite. On the other hand, if D_A has a path from j^- to j^+ for each $j \in C^+$ then it follows from Proposition 3.2 that the maximum is finite. So, the finiteness of the maximum can be tested during the computation of x^*. If the maximum is finite then by Theorem 4.1, x^* is an optimum solution. ∎

As a consequence of Corollary 4.1 we have that $\max\{c^T x \mid Ax \leq b\}$ with A satisfying (4.1) and (4.3) can be solved in $O(nm + n^2 \log n)$ time (cf. Fredman & Tarjan [1987]).

Remark 4.1. From the correspondence between solutions of $Ax \leq b$ and potentials on D_A given by (3.4) and (3.5) we see that

$$\begin{aligned} \max\{c^T x \mid Ax \leq b\} &= \max\{d^T \pi \mid \pi^T N \leq l\} \\ &= \min\{l^T f \mid Nf = d,\ f \geq 0\} \\ &= \min\{b^T y \mid y^T A = c,\ y \geq 0\}, \end{aligned}$$

where N is the incidence matrix of $D_A = (V, E)$, d is a V-vector so that $d_{i^+} = -d_{i^-} = c_i$ and l is an E-vector so that $l(u,v) = \text{length}_{D_A}(u,v)$ for each arc (u,v) of D_A. The problem $\min\{l^T f \mid Nf = d,\ f \geq 0\}$ is the uncapacitated minimum cost flow problem (or the Transshipment problem) and it can be solved in time $O(n \log n T_{ss}(m,n))$ (cf. Orlin [1993]). Charnes & Raike [1966] applied a variant of Dijkstra's algorithm to solve the generalized flow problem for networks with one source and one sink when the "multipliers" of each arc are at most one.

4.3. Finding an integral maximal element

Suppose $Ax \leq b$ has an integral solution. Without loss of generality we may

assume that 0 is a solution of $Ax \leq b$ (i.e. $b \geq 0$). (Indeed, by Theorem 3.3, an integral solution y^* of $Ax \leq b$ can be found in $O(T_{ss}(m,n))$ time. Now, x^* is a maximal element of $\{x \mid Ax \leq b\}$ if and only if $x^* - y^*$ is a maximal element of $\{x \mid Ax \leq b - Ay^*\}$.)

Lemma 4.3 *Let $j \in \{1,\ldots,n\}$ be so that j^+ is reachable from j^- in D_A. Then $\text{distance}_{D_A}(j^-, j^+)$ is even.*

Proof. Let $P_{j^-j^+}$ be a shortest path from j^- to j^+. By condition (4.3) we have that $P_{j^-j^+}$ must contain an arc (k^-, k^+) for some $k \in \{1,\ldots,n\}$. By the symmetry of D_A we may assume that $P_{j^-j^+} = P^- \cdot (k^-, k^+) \cdot P^+$ where P^- is a path from j^- to k^- traversing only vertices in V^- and P^+ is a $k^+ - j^+$-path traversing only vertices in V^+ so that (u^-, v^-) is an arc of P^- if and only if (v^+, u^+) is an arc of P^+. Thus $\text{distance}_{D_A}(j^-, j^+) = 2 \cdot \text{length}_{D_A}(P^-) + \text{length}_{D_A}(k^-, k^+)$ which is even. ∎

From Lemma 4.3 it follows that if y^* is an integral solution of $Ax \leq b$ then the maximal element x^* of $\{x \mid Ax \leq b\}$ defined in (4.4) is integral.

Corollary 4.2 *Let b be an integral m-vector and let A be a matrix satisfying (4.1) and (4.3). Then $\{x \mid Ax \leq b\}$ has an integral maximal element, provided that $Ax \leq b$ has an integral solution. Moreover, an integral maximal element of $\{x \mid Ax \leq b\}$ can be found in $O(nT_{ss}(m,n))$ time.* ∎

Corollary 4.3 *Let b be an integral m-vector and let A be a matrix satisfying (4.1) and (4.3). Then $\max\{c^T x \mid Ax \leq b,\ x \text{ integral}\}$ can be solved in $O(nT_{ss}(m,n))$ time for each $c \geq 0$.* ∎

It follows from Corollary 4.3 that $\max\{c^T x \mid Ax \leq b,\ x \text{ integral}\}$ with A satisfying (4.1) and (4.3) can be solved in $O(nm + n^2 \log n)$ time (cf. Fredman & Tarjan [1987]).

Chapter 5

Dynamic programming and shortest disjoint paths in planar graphs

5.1. Introduction

In transmission of messages in a communication network one is interested in routing messages from origin stations to destination stations. Constraints in sending a message are of two types: (*i*) at any time, a station can participate in at most one transmission, and (*ii*) a station can transmit only to an adjacent station. Two stations that are not directly connected can communicate by using a series of stations that form a path from a origin station to a destination station. Associated with each transmission line there is a fixed delay (or cost). The total delay along the communication path between an origin station and a destination station is the sum of the delay on each of the transmission lines on the path. There are at least two possible optimization problems for this type of routing problem: (*i*) the *min-sum problem* consisting of minimizing the sum of the delays of the routes (minimum cost routing); (*ii*) the *min-max problem* consisting of minimizing the time at which the last transmission is completed. (Another possible motivation for this kind of min-max requirement is the existence of noise in transmission lines which increases with the length of the path between the origin station and destination station—this is also a possible requirement to VLSI-routing, where the length of the longest path connecting two nets dictates the speed of the chip.) A communication network can be modeled as an undirected graph in which vertices represent the stations, edges represent the transmission lines and the length of an edge represents the delay (or cost) on transmitting a message though the corresponding transmission line. (Thus a shortest path in this graph between two vertices corresponds to a path in the communication network for which the sum of delays in minimum.) The problem of determining disjoint routings of messages in a communication network translates into finding vertex disjoint paths in this undirected graph model between pairs of specific vertices.

Most of the research on disjoint paths problems has concentrate on whether or not a given graph has a collection of disjoint paths between specific

terminals. In the present chapter we apply dynamic programming techniques in order to find a collection of "short" disjoint paths between specific endpoints in a planar graph. These problems turns out to be much harder than just finding disjoint paths; even simple cases where finding disjoint paths is easy are $\mathcal{NP}$-hard for finding short disjoint paths. More specifically, we use dynamic programming techniques in order to show the polynomial or pseudo-polynomial time solvability of some short disjoint paths problems in planar graphs with the following special structure: each terminal is a vertex along the unbounded face. The technique we apply here consists of reformulating these disjoint paths problems in terms of finding a shortest path in a certain (large) acyclic digraph. The *integer knapsack problem* is a well-known example of an important combinatorial optimization problem that can be reformulated in these terms. In particular, from this reformulation one derives a pseudo-polynomial time algorithm for the integer knapsack problem (see, e.g., Papadimitriou & Steiglitz [1982: Ch. 16]). Actually, integer linear programming itself can be viewed as a shortest path problem in a large acyclic graph (cf. Wolsey [1974]). This yields a pseudo-polynomial time algorithm for integer linear programming if the number of constraints is fixed (cf. Papadimitriou [1981]; see also Schrijver [1986: Ch. 18]).

The organization of this chapter is as follows. In Section 5.2 we make some notes on the "shortest disjoint paths problem" (min-sum problem). Next, in Section 5.3 we discuss the "length-bounded disjoint paths problem" (min-max problem). In Section 5.4 we show that the length-bounded disjoint paths problem (stated as a decision problem) is $\mathcal{NP}$-complete in the strong sense for nonfixed k. In Section 5.5, we consider an auxiliary problem. Finally, in Sections 5.6 and 5.7 we show the pseudo-polynomial time solvability for particular instances of the two and three length-bounded disjoint paths problem, respectively; we do this by reformulating the problem in terms of the auxiliary problem considered in Section 5.5

5.2. Minimizing sum of lengths

Consider the following *shortest disjoint paths* problem:

given : – an undirected planar graph $G = (V, E)$, embedded in $\mathbb{R}^2$;
– pairs $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ of vertices on the boundary of G;
– a length function $l : E \to \mathbb{Z}_+$;

find : – pairwise vertex-disjoint paths $P_1, \ldots, P_k$ in G so that for each $i = 1, \ldots, k$, P_i is an $r_i - s_i$-path and so that (5.1)

$$\sum_{i=1}^{k} \text{length}_l(P_i) \text{ is as small as possible.}$$

We denote this problem by $\text{SDP}(G, \{r_1, s_1\}, \ldots, \{r_k, s_k\})$.

If the vertices $r_1, \ldots, r_k, s_k, \ldots, s_1$ occur in this order when following

the boundary of G clockwise, then problem (5.1) can be seen as a particular case of the min-cost flow problem. Indeed, from G we construct a directed graph D by splitting each vertex v of G into two vertices v^+ and v^- joined by an arc (v^-, v^+) of cost zero. An edge (v, w) of G becomes arcs (v^+, w^-) and (w^+, v^-) of D, both of cost $l(\{v, w\})$. In addition D has vertices r, s and arcs $(r, r_i^-), (s_i^+, s)$, $i = 1, \ldots, k$, of cost zero. Solving (5.1) is equivalent to finding a maximum $r - s$-flow of minimum cost in D, with each arc having capacity one. Hence, in this particular case (5.1) can be solved in polynomial time (see, e.g., Ahuja, Magnanti & Orlin [1993: Ch. 9 and 10]).

A graph $G = (V \cup \{c\}, E)$ is called a *wheel* if $G - c$ is a circuit and $\{v, c\} \in E$ for each $v \in V$. Grötschel, Martin & Weismantel [1993] have shown that if $G = (V \cup \{c\}, E)$ is a wheel and $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the circuit $G - c$ then the edge-disjoint version of (5.1) can be solved in polynomial time. Moreover, Grötschel, Martin & Weismantel gave a complete description of the *path packing polytope* (the convex hull of incident vectors of sets $E' \subseteq E$, such that $G[E']$ is a packing of edge-disjoint $r_i - s_i$-paths in G).

Using dynamic programming we show that also in the following case the shortest disjoint path problem (5.1) can be solved in polynomial time:

k is fixed and the vertices $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the boundary of G clockwise. (5.2)

We assume that problem (5.1) is feasible (this can be tested in linear time) and that G is 2-connected.

We shall use the following notation. If P is a path and u, v are vertices in P then $P(u, v)$ is the subpath of P connecting u to v. Q_i will denote a shortest $r_i - s_i$-path and R_i will denote the subgraph of G induced by the vertices in the closed region bounded by the path Q_i and the path on the boundary of G clockwise from r_i to s_i $(i = 1, \ldots, k)$. Let $Q_{i,j} := R_i \cap R_j$. By shortcut arguments we may assume that for $i \neq j$, $Q_{i,j}$ is either a path or empty. Figure 5.1 illustrates the notation. In the figure, P_1, P_2, P_3 are paths of an optimal solution.

Let G, $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ and $l : E \to \mathbb{Z}_+$ be an instance of the shortest disjoint paths problem (5.1) with the property that $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the boundary of G (say) clockwise. Observe that problem (5.1) has an optimal solution $P_1, \ldots, P_k$ so that P_i is entirely contained in region R_i $(i = 1, \ldots, k)$.

Consider some i, j, $i \neq j$. Denote $Q_{i,j} = (v_0, e_1, v_1, \ldots, e_d, v_d)$. We call (f, l) a *possible choice* (for (i, j)) if $f = l =$ nil or $f = v_p$ and $l = v_q$ for some p, q satisfying $0 \leq p \leq q \leq d$. We say that (f, l, f', l') is a *feasible choice* (for (i, j)) if (f, l) and (f', l') are possible choices and if $f \neq nil \neq f'$ implies $\{f, l\} \cap \{f', l'\} = \emptyset$. For each feasible choice (f, l, f', l') let

$$G_{i,j,f,l,f',l'} \quad := \quad G[V(R_i) \setminus (V(Q_{i,j}) \setminus V(Q_{i,j}(f, l)))]$$

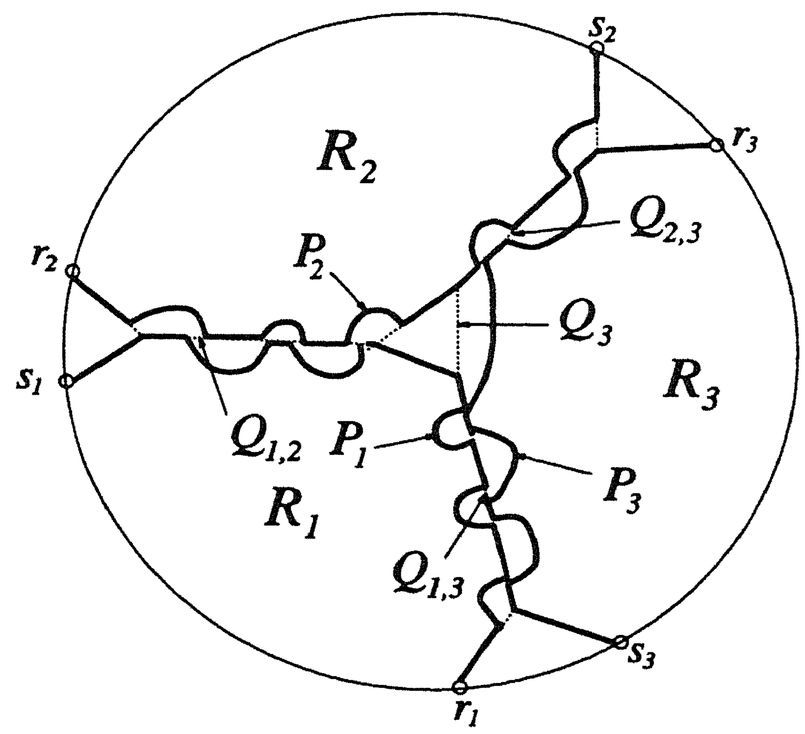


Figure 5.1: Illustration of the definitions.

$$\cup\, G[V(R_j) \setminus (V(Q_{i,j}) \setminus V(Q_{i,j}(f', l')))],$$

where $Q_{i,j}(\text{nil},\text{nil}) := \emptyset$. Finally, we say that a sequence

$$((f_{i,j}, l_{i,j}, f'_{i,j}, l'_{i,j}) : i, j = 1, \ldots, k,\ i < j)$$

is an *acceptable choice* provided that $(f_{i,j}, l_{i,j}, f'_{i,j}, l'_{i,j})$ is a feasible choice for each (i, j), $i, j = 1, \ldots, k, i < j$.

Our dynamic programming approach is based on the following optimality criterion:

> Let $P_1, ..., P_k$ be an optimal solution to (5.1) and let f_i, l_i, f_j, l_j be the first and last vertices of $Q_{i,j}$ in P_i and in P_j, respectively, for some $i, j, 1 \leq i < j \leq k$. Then $P_i(f_i, l_i), P_j(f_j, l_j)$ is an optimal solution to $\text{SDP}(G_{i,j,f_i,l_i,f_j,l_j}, \{f_i, l_i\}, \{f_j, l_j\})$ (see Figure 5.2).

The algorithm consists of first computing, for each $(i, j), i, j = 1, \ldots, k$, $i < j$, and for each feasible choice (f, l, f', l') for (i, j), a solution $P_{i,j,f,l,f',l'}$, $P_{j,i,f,l,f',l'}$ to $\text{SDP}(G_{i,j,f,l,f',l'}, \{f, l\}, \{f', l'\})$, where, $P_{i,j,\text{nil},\text{nil},f',l'} := \emptyset$ and $P_{j,i,f,l,\text{nil},\text{nil}} := \emptyset$). Now, we enumerate all acceptable choices

$$A \quad := \quad ((f_{i,j}, l_{i,j}, f'_{i,j}, l'_{i,j}) : i, j = 1, \ldots, k, i < j)$$

(so, $(f_{i,j}, l_{i,j}, f'_{i,j}, l'_{i,j})$ is a feasible choice for (i, j) $i, j = 1, \ldots, k, i < j$) and we compute $P_1^A, \ldots, P_k^A$, where P_i^A is a shortest $r_i - s_i$-path in

$$G[(V(R_i) \setminus V(Q_{i,j}))\ \cup\ (\cup_{i \neq j} V(P_{i,j,f_{i,j},l_{i,j},f'_{i,j},l'_{i,j}}))]\quad (i = 1, \ldots, k),$$

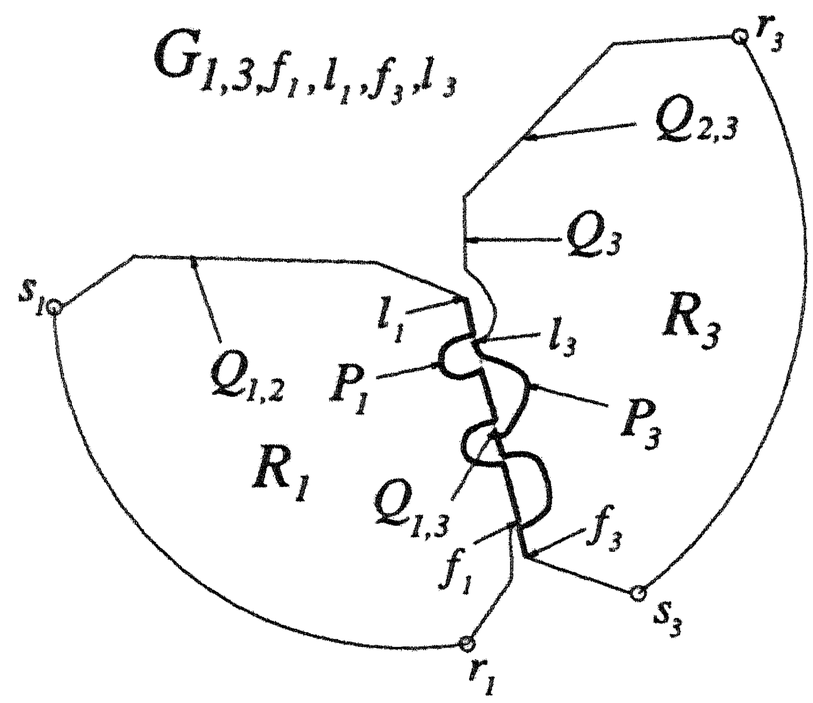


Figure 5.2: Illustration of the optimality condition for the shortest disjoint paths problem (5.1).

if there exists any.

The algorithm returns, for some acceptable choice A^*, vertex-disjoint paths $P_1^{A^*}, \ldots, P_k^{A^*}$ so that,

$$\sum_{i=1}^{k} \text{length}(P_i^{A^*}) = \min\{\sum_{i=1}^{k} \text{length}(P_i^A) \mid A \text{ is an acceptable choice}\}.$$

Theorem 5.1 *If $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the boundary of G clockwise, then for each fixed k the shortest disjoint paths problem (5.1) can be solved in polynomial time.*

Proof. Any solution $P_1, \ldots, P_k$ to (5.1), such that P_i is a subgraph of R_i, induces an acceptable choice to problem (5.1). (For each $(i,j), i < j$, we define $(f_{i,j}, l_{i,j}, f'_{i,j}, l'_{i,j})$ as the first and last vertices of $Q_{i,j}$ in P_i and P_j, respectively. If the path $Q_{i,j}$ does not meet, say P_i then $f_{i,j} := l_{i,j} :=$ nil). Since by shortcut arguments there exists at least one such a solution, it follows that the algorithm generates and returns a solution to (5.1).

One sees that $\text{SDP}(G_{i,j,f,l,f',l'}, \{f,l\}, \{f',l'\})$ can be solved in polynomial time, as (if $f \neq \text{nil} \neq f'$) f, l, f', l' are vertices on the boundary of $G_{i,j,f,l,f',l'}$. Thus, in order to show polynomiality it remains to verify that the number of acceptable choices is polynomially bounded. Indeed,

$$|\{Q_{i,j} \mid i < j \text{ and } Q_{i,j} \neq \emptyset\}| = O(k^2)$$

and there exist $O(n^4)$ feasible choices for each (i,j). Hence, there are $O(n^{4k^2})$ acceptable choices. ∎

Remark 5.1 Using similar techniques one can prove that if $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the boundary of G clockwise then the edge-disjoint version of problem (5.1) is polynomially solvable for each fixed k. It can also be proved that if in addition $|i-j| > 1$ implies $Q_{i,j} = \emptyset$ then (5.1) can be solved in polynomial time (k need not to be fixed).

5.3. Minimizing maximum length

Consider the following *length-bounded disjoint paths problem*::

given : – an undirected planar graph $G = (V, E)$;
– pairs $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ of vertices of G;
– positive integers $b_1, \ldots, b_k$;
– a length function $l : E \to \mathbb{Z}_+$;

find : – pairwise vertex-disjoint paths $P_1, \ldots, P_k$ so that for each $i = 1, \ldots, k$, P_i is an $r_i - s_i$-path and so that (5.3)

$$\text{length}_l(P_i) \leq b_i, \ (i = 1, \ldots, k).$$

This problem seems to be considerably harder than the shortest disjoint paths problem (5.1) discussed in the previous section. While problem (5.1) is polynomially solvable for nonfixed k and $r_1 = \ldots = r_k$ (via a min-cost flow algorithm, see, e.g. Ahuja, Magnanti & Orlin [1993: Ch. 9 and 10])) problem (5.3) is easily proved to be $\mathcal{NP}$-complete for planar graphs with $k = 2$, $r_1 = r_2$ and $s_1 = s_2$ (cf. Itai, Perl & Shiloach [1982]). Li, McCormick & Simchi-Levi [1990] showed that problem (5.3) is strongly $\mathcal{NP}$-complete for $k = 2$. Li, McCormick & Simchi-Levi, extending techniques in Perl & Shiloach [1978], described also a pseudo-polynomial time algorithm for the problem of finding k (fixed) length-bounded vertex (arcs) disjoint paths from a vertex r to a vertex s in acyclic directed graph.

Itai, Perl & Shiloach [1982] considered the related problem of finding in an undirected graph the maximum number of vertex (edge) disjoint paths of length (bounded by) $L \in \mathbb{Z}_+$ connecting two vertices r and s. They showed that for $L \geq 4$ ($L \geq 5$ for the bounded case) these problems are all $\mathcal{NP}$-complete.

In the sequel we focus on the following restrictive version of problem (5.3):

given : – an undirected planar graph $G = (V, E)$ embedded in $\mathbb{R}^2$;
– r, s distinct vertices of G adjacent to the unbounded face;
– positive integers $b_1, \ldots, b_k$;
– a length function $l : E \to \mathbb{Z}_+$;

find : – pairwise internally vertex-disjoint $r - s$-paths $P_1, \ldots, P_k$ so that (5.4)

$$\text{length}_l(P_i) \leq b_i, \ (i = 1, \ldots, k).$$

Extending ideas of these previous papers we show that:

(*i*) problem (5.4) is $\mathcal{NP}$-complete in the strong sense for nonfixed k; and
(*ii*) for $k \leq 3$ problem (5.4) can be solved in pseudo-polynomial time.

5.4. $\mathcal{NP}$-completeness

The *3-Satisfiability problem* (3SAT) consists of: *given*: a set of variables $U = \{u_1, \ldots, u_n\}$ and a set of three-literal clauses $C = \{c_1, \ldots, c_m\}$, each literal been either u_i or $\bar{u}_i$; *question*: is there a satisfying truth assignment for C? A satisfying truth assignment for C is a function $t : U \to \{\text{true}, \text{false}\}$ such that each clause of C contains at least one true literal. 3SAT was shown to be $\mathcal{NP}$-complete by Cook [1971].

Theorem 5.2 *The length-bounded disjoint paths problem (5.4) is $\mathcal{NP}$-complete in the strong sense.*

Proof. It is easy to see that problem (5.4) belongs to $\mathcal{NP}$. We transform 3SAT to problem (5.4) for instances where $b_1 = \ldots = b_k = 2n + 4$ and $l(e) = 1, 2$ or 3 for each $e \in E$.

Let $U = \{u_1, \ldots, u_n\}$ be a set of variables and $C = \{c_1, \ldots, c_m\}$ be the set of clauses in an arbitrary instance of 3SAT. Corresponding to this instance of 3SAT we define a graph $G = (V, E)$ and a length function $l : E \to \mathbb{Z}_+$ as follows:

$$\begin{aligned}
V &:= \{r, s\} \cup \{(i,j) \in \mathbb{Z}_+^2 \mid i \leq 4n + 2m; \\
&\qquad j \leq 4n \text{ and } i + j \equiv 1 \pmod 2\}; \\
E &:= \{\{(i,j),(k,l)\} \mid (i,j),(k,l) \in V; \\
&\qquad (i,j) \neq (k,l) \text{ and } ||(i-k, j-l)||_\infty = 1\} \\
&\cup \{\{r,(i,0)\} \mid i = 1, 3, \ldots, 4n + 2m - 1\} \\
&\cup \{\{(i,4n), s\} \mid i = 1, 3, \ldots, 4n + 2m - 1\};
\end{aligned}$$

$$l(e) := \begin{cases}
3 & \text{if} \quad e = \{(2i-1, 4i-4), (2i-2, 4i-3)\} \\
& \text{or} \quad e = \{(2i-1, 4i-2), (2i-2, 4i-1)\} \\
& \text{or} \quad e = \{(4n+2m-2i+1, 4i-4), (4n+2m-2i+2, 4i-3)\} \\
& \text{or} \quad e = \{(4n+2m-2i+1, 4i-2), (4n+2m-2i+2, 4i-1)\} \\
& \qquad \text{for } i = 1, \ldots, n; \\
2 & \text{if} \quad e = \{(2(n+j)-1, 4i-4), (2(n+j), 4i-3)\} \text{ and } u_i \in c_j \\
& \text{or} \quad e = \{(2(n+j)-1, 4i-4), (2(n+j)-2, 4i-3)\} \text{ and } \bar{u}_i \in c_j \\
& \qquad \text{for } i = 1, \ldots, n; j = 1, \ldots, m; \\
1 & \qquad \text{otherwise}
\end{cases}$$

Figure 5.3 shows the graph associated with

$$C = \{\{\bar{u}_1, \bar{u}_2, u_3\}, \{u_1, \bar{u}_2, \bar{u}_3\}, \{\bar{u}_1, u_2, \bar{u}_3\}, \{u_1, u_2, u_3\}\}.$$

In the figure the number inside the circle next to an edge corresponds to the length of that edge. (The length-one edges have no circle close to them on the figure, to avoid overcrowding.)

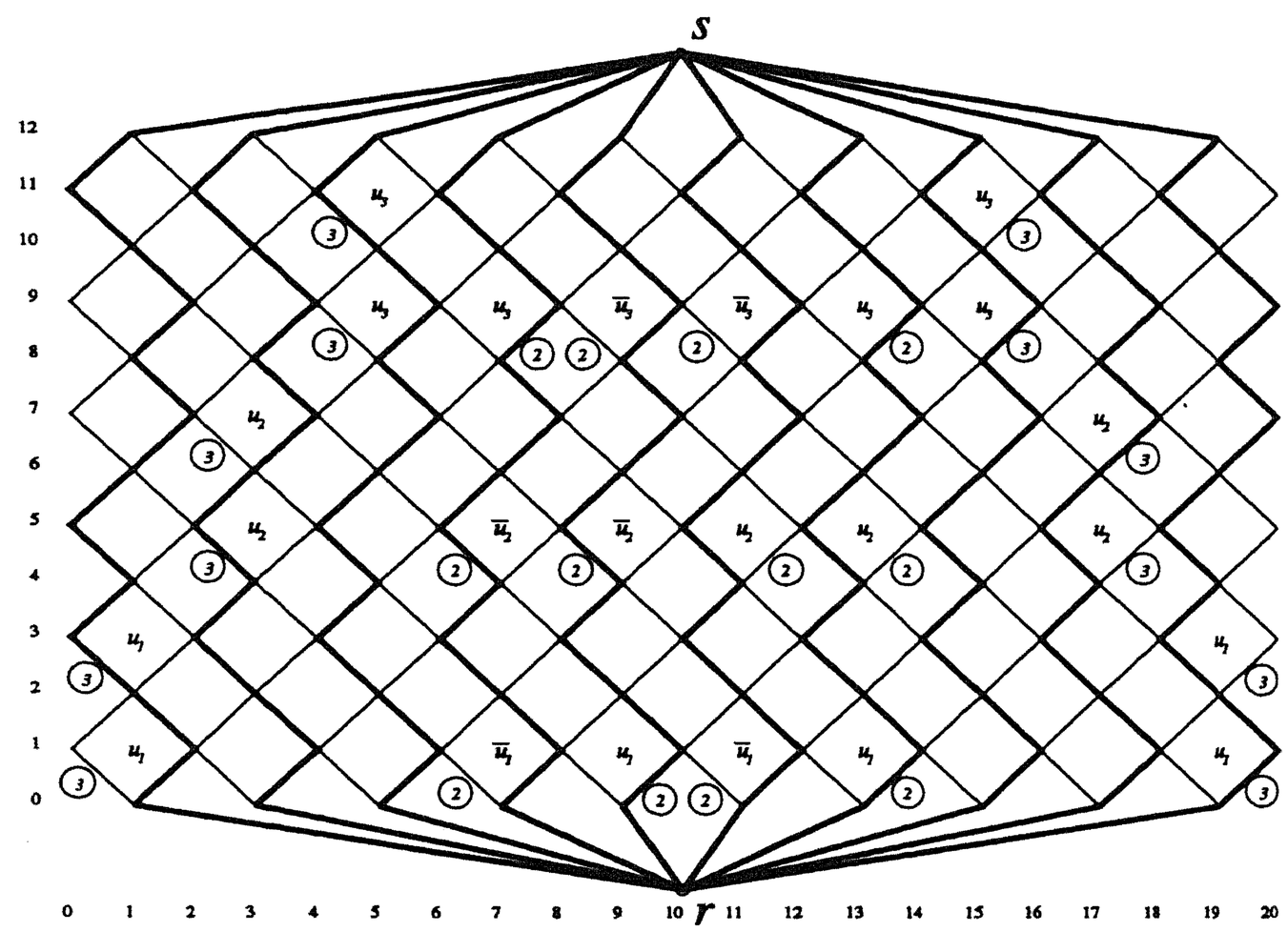


Figure 5.3: Instance of the maximum length-bounded disjoint paths problem associated with the clause $C = \{\{\bar{u}_1, \bar{u}_2, u_3\}, \{u_1, \bar{u}_2, \bar{u}_3\}, \{\bar{u}_1, u_2, \bar{u}_3\}, \{u_1, u_2, u_3\}\}$.

One easily sees that given an instance of 3SAT the corresponding graph G can be constructed in polynomial time and that it can be embedded in $\mathbb{R}^2$ so that r and s are adjacent to the unbounded face. Now, we claim that C is satisfiable if and only if the instance $(G = (V, E); \{r, s\}; b_1 = \ldots = b_k = 2n + 4; l)$ of problem (5.4) is feasible.

Suppose $P_1, \ldots, P_{2n+m}$ form a feasible solution of the instance constructed above. The path P_{n+j} is the path associated with clause c_j of C $(j = 1, \ldots, m)$ and the paths P_i and $P_{2n+m-i+1}$ are the paths associated with the variable u_i $(i = 1, \ldots, n)$. First we observe that the path P_i must traverse all vertices in $\{(2i-1, j) \in V \mid j = 0, 2, \ldots, 2n\}$ $(i = 1, \ldots, 2n + m)$. Moreover, as each path has length at least $2n + 2$ and at most $2n + 4$, the subpaths of P_i and $P_{2n+m-i+1}$ connecting $(2i-1, 4i-4)$ to $(2i-1, 4i)$ and $(4n + 2m - 2i + 1, 4i - 4)$ to $(4n + 2m - 2i + 1, 4i)$, respectively, must be as shown in either Figure 5.4(a) or Figure 5.4(b) $(i = 1, \ldots, n)$. (In the figure the

paths are represented by thicker lines.)

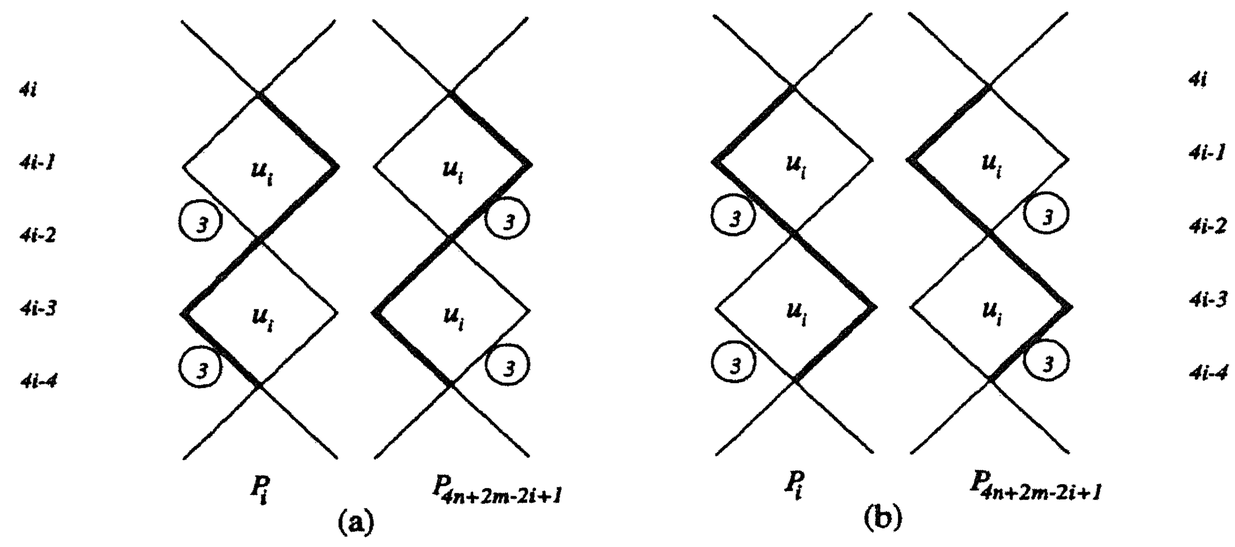


Figure 5.4: Figure (a) shows the subpaths corresponding to the assignment $t(u_i) :=$ true and Figure (b) shows the subpaths corresponding to the assignment $t(u_i) :=$ false.

We define a truth assignment $t : \{u_1, \ldots, u_n\} \to \{\text{true}, \text{false}\}$ to C as follows

$$t(u_i) := \begin{cases} \text{true} & \text{if paths } P_i((2i-1, 4i-4), (2i-1, 4i)) \text{ and} \\ & P_{2n+m-i+1}((4n+2m-2i+1, 4i-4), (4n+2m-2i+1, 4i)) \\ & \text{are as shown in figure 5.4(a);} \\ \text{false} & \text{if paths } P_i((2i-1, 4i-4), (2i-1, 4i)) \text{ and} \\ & P_{2n+m-i+1}((4n+2m-2i+1, 4i-4), (4n+2m-2i+1, 4i)) \\ & \text{are as shown in figure 5.4(b).} \end{cases}$$

The truth assignment corresponding to the paths in Figure 5.3 is $t(u_1) =$ false, $t(u_2) =$ true and $t(u_3) =$ false.

Consider the path P_{n+j} associated with the clause c_j of C. Since P_{n+j} has length at most $2n+4$, this path can not use the three length-two edges corresponding to the three literals in c_j (length-two edges having a point with first coordinate $2(n+j)-1$ as one of its end vertices). The length-two edge which was not used by P_{n+j} corresponds to a true literal in the clause c_j.

Conversely, if C is satisfiable and t is one of its satisfying truth assignment then we can use $t(u_i)$ to define the subpaths of P_i and $P_{2n+m-i+1}$ connecting $(2i-1, 4i-4)$ to $(2i-1, 4i)$ and $(4n+2m-2i+1, 4i-4)$ to $(4n+2m-2i+1, 4i)$, respectively, as shown in either figure 5.4(a) if $t(u_i) =$ true or in Figure 5.4(b) if $t(u_i) =$ false. There exists a unique way to extend these subpaths to pairwise internally vertex-disjoint $r-s$-paths $P_1, \ldots, P_{2n+m}$ in G. One easily checks that each one of these paths has length at most $2n+4$. ∎

Remark 5.2. The length-bounded disjoint path problem (5.4) remains $\mathcal{NP}$-

complete in the strong sense if one replaces the condition length$(P_i) \leq b_i$ by length$(P_i) = b_i$. (Using the same construction as used in Theorem 5.2 one can transform *One-in-three 3SAT* (see Garey & Johnson [1979]) to this new problem.) Problem (5.4) also remains $\mathcal{NP}$-complete in the strong sense if instead of internally vertex-disjoint paths one is interested in edge-disjoint paths. (We could by "decontracting" each vertex $v \neq r, s$ of degree 4 transform the graph constructed in the theorem in a graph where each vertex $v \neq r, s$ has maximum degree 3.)

5.5. Auxiliary problem

In the next section we reduce problem (5.4) for $k = 2, 3$ to the following problem:

$$\begin{array}{ll} \textit{given}: & \text{– a directed acyclic graph } D = (W, A); \\ & \text{– distinct vertices } R \text{ and } S \text{ of } D; \\ & \text{– positive integers } b_1, \ldots, b_k; \\ & \text{– length functions } l_1, \ldots, l_k : A \to \mathbb{Z}_+; \\ \textit{find}: & \text{– a directed } R-S\text{-path } P = (R, a_1, \ldots, a_t, S) \text{ such that} \\ & \qquad \text{length}_{l_i}(P) \leq b_i, \quad (i = 1, \ldots, k), \\ & \qquad \text{where length}_{l_i}(P) = \sum_{j=1}^{t} l_i(a_j). \end{array} \tag{5.5}$$

In this section we observe how problem (5.5) can be solved in $O(ML^k)$ time and $O(NL^k)$ space, where $N = |W|, M = |A|$ and $L = \max_{1 \leq i \leq k} b_i$.

We use here the same idea used by Li, McCormick & Simchi-Levi [1990] in order to find two length-bounded disjoint paths in an acyclic directed graph in pseudo-polynomial time. Namely, we shall keep with each vertex v of D a table T_v with entries indexed by k-tuples $(d_1, \ldots, d_k)$, $0 \leq d_i \leq L$ for $i = 1, \ldots, k$. (So T_v has $(L+1)^k$ entries.) During the execution of the algorithm an entry contains either nil or an ordered pair $(u, (d'_1, \ldots, d'_k))$ for some $u \in W$ and $(d'_1, \ldots, d'_k) \in \mathbb{Z}_+^k$ so that $(u, v) \in A$ and $d_i = d'_i + l_i(u, v)$ $(i = 1, \ldots, k)$.

Let $v_1, \ldots, v_N$ be an ordering of the vertices of D such that $(v_i, v_j) \in A$ implies $i < j$. (The so-called *topological ordering* of the vertices of D; it can be found in $O(M)$ time.) For our proposes we may assume that $v_1 = R$ and $v_N = S$.

The algorithm begins by setting the entry $(0, \ldots, 0)$ of the table T_R to $(R, (0, \ldots, 0))$ and all others entries of T_R and all entries of all others tables to nil. For $i = 1, \ldots, N$ we iterate the following:

> for each vertex u so that $a := (u, v_i) \in A$ and for each entry $(d_1, \ldots, d_k)$ of T_u with contents different of nil:
> set the entry $(d_1 + l_1(a), \ldots, d_k + l_k(a))$ of T_{v_i} to $(u, (d_1, \ldots, d_k))$.

If any entry $(d_1, \ldots, d_k)$ of T_S, $(d_i = 0, \ldots, b_i;\ i = 1, \ldots, k)$, has its contents different of nil, say equal to $(u, (d'_1, \ldots, d'_k))$, then there exists an $R - S$-path $(R = v_0, a_1, v_1, \ldots, v_t, a_t, u, (u, S), S)$ in D, $v_i \in W$ $(i = 0, \ldots, t)$ $a_i \in A$ $(i = 1, \ldots, t)$, so that $\sum_{j=1}^{t} l_i(a_j) + l_i(u, S) = d'_i + l_i(u, S) = d_i \leq b_i$ $(i = 1, \ldots, k)$. Otherwise the given instance of problem (5.5) is infeasible.

The algorithm essentially enumerates all possible length vectors

$$(d_1, \ldots, d_k) \quad (d_i = 0, \ldots, L;\ i = 1, \ldots, k),$$

and for each one of them finds a $R - S$-path P so that $\text{length}_{l_i}(P) = d_i$ for each i, if any. By induction on the number of iterations of the algorithm one can show that at end of iteration i the following holds:

> the entry $(d_1, \ldots, d_k)$ of T_{v_i} contains $(u, (d'_1, \ldots, d'_k))$ for some $u \in W$ and $(d'_1, \ldots, d'_k) \in \mathbb{Z}_+^k$ if and only if there exists a path
>
> $$(R, a_1, \ldots, a_t, u, (u, v_i), v_i)$$
>
> in D such that
>
> $$\sum_{j=1}^{t} l_i(a_j) + l_i(u, v_i) = d'_i + l_i(u, v_i) = d_i \ (i = 1, \ldots, k).$$

This implies the correctness of the algorithm. In order to actually construct an $R - S$-path at the end of the algorithm we proceed as follows. Suppose the contents of the entry $(d_1, \ldots, d_k)$ of T_S is not nil. Define $v_1 := S$, $d_{1,j} := d_j$ $(j := 1, \ldots, k)$, set i to 1 and iterate the following until $v_i = R$

> increment i and define $v_i := u$; $d_{i,j} := d_{i-1,j} - l_i(u, v_{i-1})$, where u is the vertex in the entry $(d_{i-1,1}, \ldots, d_{i-1,k})$ of $T_{v_{i-1}}$.

Eventually, if $v_{t+1} = R$ then

$$(R = v_{t+1}, a_t, v_{t-1}, \ldots, a_1, v_1 = S),$$

where $a_i = (v_{i+1}, v_i)$ $(i = 1, \ldots, t)$, is an $R - S$-path satisfying $\sum_{j=1}^{t} l_i(a_j) = d_i$.

The space required by the algorithm is clearly $O(NL^k)$. The amount of time needed to initialize all tables is $O(NL^k)$ and the time necessary to scan vertex v_i during iteration i is $O(d^+(v_i)L^k)$, where $d^+(v_i)$ is the indegree of v_i. Hence, the total time required by the algorithm is $O(ML^k)$.

Proposition 5.1 *Problem (5.5) can be solved in $O(ML^k)$ time and $O(NL^k)$ space.* ∎

5.6. Case $k = 2$

In this section we show that problem (5.4) can be solved in pseudo-polynomial time for $k = 2$. In order to do so we reduce problem (5.4) for $k = 2$ to

problem (prob2). (Note that the case $k = 1$ can be solved by a single-source shortest path algorithm.)

Let $G = (V, E)$, $r, s \in V$, $b_1, b_2 \in \mathbb{Z}_+$ and $l : E \to \mathbb{Z}_+$ be an instance of problem (5.4). Before proceeding we need some definitions (we shall also use these definitions in the next section). Let Q be a shortest $r - s$-path in G and suppose that Q follows the vertices $r = v_0, v_1, \ldots, v_t = s$. For $v, w \in V(Q)$ we write $v \preceq w$ if $v = v_j$ and $w = v_k$ for some $j \leq k$. We write $v \prec w$ if $v \preceq w$ and $v \neq w$. (So, $\preceq$ is a partial ordering on $V(Q)$.) We denote by R_1 the subgraph of G induced by the vertices in the close region bounded by the path Q and the path on the boundary of G clockwise from s to r. By R_2 we denote the subgraph $G \setminus (V(R_1) \setminus V(Q))$. For $u, v \in V(Q)$ and $i = 1, 2$ let $P(i; u, v)$ be a shortest $u - v$-path in the graph $R_i \setminus (V(Q) \setminus \{u, v\})$. Figure 5.5 illustrates the definitions.

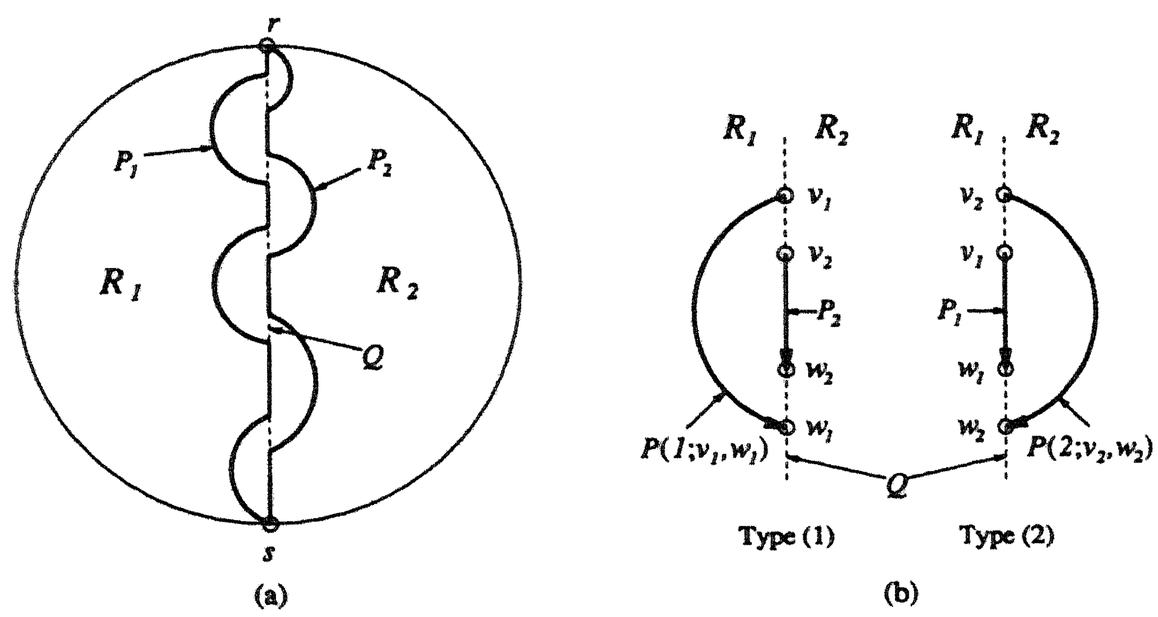


Figure 5.5: Illustration of the definitions.

Make an auxiliary digraph $D_2(G, Q) = (W_2, A_2)$ with lengths $l_1, l_2 : A_2 \to \mathbb{Z}_+$ on its arcs as follows. The vertex set W_2 consists of couples (v_1, v_2) of vertices of Q. In D_2 there is an arc from (v_1, v_2) to (w_1, w_2) if and only if for some $q \in \{1, 2\}$ one of the following conditions hold:

$$
\begin{array}{ll}
(i) & v_q \prec v_{3-q} \preceq w_{3-q} \prec w_q; \\
(ii) & v_1 = v_2 = r \prec w_{3-q} \prec w_q; \\
(iii) & v_q \prec v_{3-q} \prec w_2 = w_1 = s.
\end{array}
$$

In addition D_2 has two parallel arcs d_1 and d_2 from (r, r) to (s, s). We say that $a \in A_2$ is an arc of type (t), $t \in \{1, 2\}$ if $a = d_t$ or a fulfills either condition (i), (ii) or (iii) above for $q = t$. If $a := ((v_1, v_2), (w_1, w_2)) \in A_2$ is an arc of type (t) then we define:

$$l_t(a) := \text{length}_l(P(t; v_t, w_t)) \quad \text{and} \quad l_{3-t}(a) := \text{length}_l(Q(v_{3-t}, w_{3-t})).$$

(See Figure 5.5(b).)

The following holds:

Lemma 5.1 *G has two internally vertex-disjoint $r-s$-paths P_1, P_2 so that $\text{length}_l(P_i) \leq b_i$ $(i = 1,2)$ if and only if D_2 has an $(r,r)-(s,s)$-path P so that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1,2)$.*

Proof. Suppose P_1 and P_2 exist. We may assume that P_i is contained in the region R_i $(i = 1,2)$. Moreover, we may assume that $u, v \in V(P_i) \cap V(Q)$ and $V(P_{3-i}) \cap V(Q(u,v)) = \emptyset$ implies that $P_i(u,v) = Q(u,v)$. (So, we may assume that P_1 and P_2 "look like" the paths in the Figure 5.5(a).)

Suppose $P_i = Q$. Then $P := ((r,r), d_{3-i}, (s,s))$ is an $(r,r)-(s,s)$-path in D_2 so that $\text{length}_{l_i}(P) \leq \text{length}_l(P_i) \leq b_i$ $(i = 1,2)$. Thus, we may assume $P_i \neq Q$ $(i = 1,2)$. Let $\{v_{1,1}, v_{2,1}\}, \ldots, \{v_{1,d-1}, v_{2,d-1}\}$ be pairs of end points of maximal subpaths of Q whose internal vertices are used neither by P_1 nor by P_2. We may assume $v_{1,1} \prec v_{2,1} \preceq v_{2,2} \prec v_{1,2} \preceq v_{1,3} \prec v_{2,3} \ldots$. Let $a_i := ((v_{1,i-1}, v_{2,i-1}), (v_{1,i}, v_{2,i}))$ $(i = 1, \ldots, d)$, where $(v_{1,0}, v_{2,0}) = (r,r)$ and $(v_{1,d}, v_{2,d}) = (s,s)$. One easily checks that $a_i \in A_2$ $(i = 1, \ldots, d)$ and $P := ((r,r), a_1, \ldots, a_d, (s,s))$ is an $(r,r)-(s,s)$-path in D_2 fulfilling the conditions of the claim.

Conversely, suppose

$$P := ((r,r) = (v_{1,0}, v_{2,0}), a_1, (v_{1,1}, v_{2,1}), \ldots, a_d, (v_{1,d}, v_{2,d}) = (s,s))$$

is an $(r,r)-(s,s)$-path in D_2 so that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1,2)$. For $i = 1,2$ and $j = 1, \ldots, d$ let

$$P_{i,j} \quad := \quad \begin{cases} P(i; v_{i,j-1}, v_{i,j}) & \text{if } a_j \text{ is an arc of type } (i); \\ Q(v_{i,j-1}, v_{i,j}) & \text{otherwise.} \end{cases}$$

We define $P_i := P_{i,1} \cdot \cdots \cdot P_{i,d}$ $(i = 1,2)$. One promptly verifies that P_1, P_2 as defined are internally vertex-disjoint $r-s$-paths (not necessarily simple) such that $\text{length}_l(P_i) = \text{length}_{l_i}(P) \leq b_i$ $(i = 1,2)$. ∎

Theorem 5.3 *The length-bounded disjoint path problem (5.4) can be solved in pseudo-polynomial time for $k = 2$.*

Proof. Lemma 5.1 implies Theorem 5.3. Indeed, by solving two all-pairs shortest path problem we can compute $\{P(i; u, v) \mid u, v \in V(Q) \text{ and } i = 1,2\}$. (For all pairs of vertices $u, v \in V(Q)$ find a shortest $u-v$-path in the graph $R_i \setminus (V(Q) \setminus \{u,v\})$ $(i = 1,2)$.) Thus, the digraph D_2 can be construct in polynomial time. D_2 is acyclic as $((v_1, v_2), (w_1, w_2)) \in A_2$ implies that (w_1, w_2) lexicographically greater than (v_1, v_2) (with respect to $\preceq$). Hence, it follows from Proposition 5.1 that an $(r,r)-(s,s)$-path P so that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1,2)$ can be found (if there exists any) in $O(|V(Q)|^4 L^2)$ time and $O(|V(Q)|^2 L^2)$ space, where $L := \max\{b_1, b_2\}$. ∎

As a consequence of Theorem 5.3 we have that the length-bounded disjoint paths problem (5.3) is polynomially solvable for G planar, $k = 2$ and r_1, s_1, r_2, s_2 on the boundary of G.

5.7. Case $k = 3$

Analogously to the case $k = 2$ we reduce problem (5.4) for $k = 3$ to problem (5.5). In this section we make use of the definitions and notation introduced in Section 5.6.

Theorem 5.4 *The length-bounded disjoint path problem (5.4) can be solved in pseudo-polynomial time for $k = 3$.*

Proof. Let $G = (V, E)$, $r, s \in V$, $b_1, b_2, b_3 \in \mathbb{Z}_+$ and $l : E \to \mathbb{Z}_+$ be an instance of problem (5.4). Moreover, let Q be a shortest $r - s$-path in G and suppose that Q follows the vertices $r = v_0, v_1, \ldots, v_t = s$, let *predeq* be a partial ordering on $V(Q)$ and let $D_2(G, Q) = (W_2, A_2)$ be the acyclic digraph as defined in the previous section. In order reduce problem (5.4) for $k =$ to problem (5.5) we make a directed acyclic auxiliary digraph $D_3 = (W_3, A_3)$ with lengths $l_1, l_2, l_3 : A_3 \to \mathbb{Z}_+$ on its arcs. We shall construct the graph D_3 in three steps.

First step—construction of D'

Make the following graph $D' = (W', A')$. The vertex set W' consists of triples (v_1, v_2, v_3) of vertices of Q. In D' there is an arc from $(v_{1,1}, v_{2,1}, v_{3,1})$ to $(v_{1,2}, v_{2,2}, v_{3,2})$ if and only if:

$$\begin{array}{ll} (i) & ((v_{1,1}, v_{2,1}), (v_{1,2}, v_{2,2})) \in A_2; \text{ and} \\ (ii) & v_{1,i} \neq v_{2,i} \Rightarrow \min\{v_{1,i}, v_{2,i}\} \prec v_{3,i} \prec \max\{v_{1,i}, v_{2,i}\} \ (i = 1, 2). \end{array}$$

We say that $((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$ is an arc of type (t) if $((r_1, r_2), (s_1, s_2)) \in A_2$ is an arc of type (t). (Observe that the graph D' is obtained from D_2 by making copies $u_1, \ldots, u_{d_u}$ of each vertex u of D_2 and $(u_i, v_j) \in A'$ if and only if $(u, v) \in A_2$.)

Before proceeding we need a few more definitions. Let

$$a := ((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$$

and suppose $((r_1, r_2), (s_1, s_2)) \in A_2$ is an arc of type (t). Suppose $P(t; r_t, s_t)$ follows the vertices $r_t = v_0, v_1, \ldots, v_d = s_t$. For each $v, w \in V(P(t; r_t, s_t) \cup Q(r_{3-t}, s_{3-t})) \cup \{r_3, s_3\}$ we write $v \preceq_a w$ if: (i) $v = v_j$ and $w = v_k$ for some $j \leq k$; (ii) $v, w \in V(Q(r_{3-t}, s_{3-t}))$ and $v \preceq w$; (iii) $v = r_3$ or $w = s_3$. As usual we write $v \prec_a w$ if $v \preceq_a w$ and $v \neq w$. (Thus, $\preceq_a$ is a partial ordering on $V(P(t; r_t, s_t) \cup Q(r_{3-t}, s_{3-t})) \cup \{r, s\}$.) We denote by R_a the subgraph of $G \setminus (\{r_1, s_1, r_2, s_2\} \setminus \{r_3, s_3\})$ induced by the vertices in the region bounded

by the paths $Q(r_t, s_t)$ and $P(t; r_t, s_t)$ which does not contain the unbounded face of G. We denote by $R_{t,a}$ the subgraph $R_t \setminus (V(R_a) \setminus V(P(t; r_t, s_t)))$. For $u, v \in V(Q(r_t, s_t) \cup P(t; r_t, s_t))$ let $P(a; u, v)$ be a shortest $u - v$ path in the graph $R_a \setminus (V(Q(r_{3-t}, s_{3-t}) \cup P(t; r_t, s_t)) \setminus \{u, v\})$ and for $u, v \in V(P(t; r_t, s_t))$ let $P(t, a; u, v)$ be a shortest $u - v$-path in the graph $R_{t,a} \setminus (V(Q \cup P(t; r_t, s_t)) \setminus \{u, v\})$. Figure 5.6(a) illustrates the definitions.

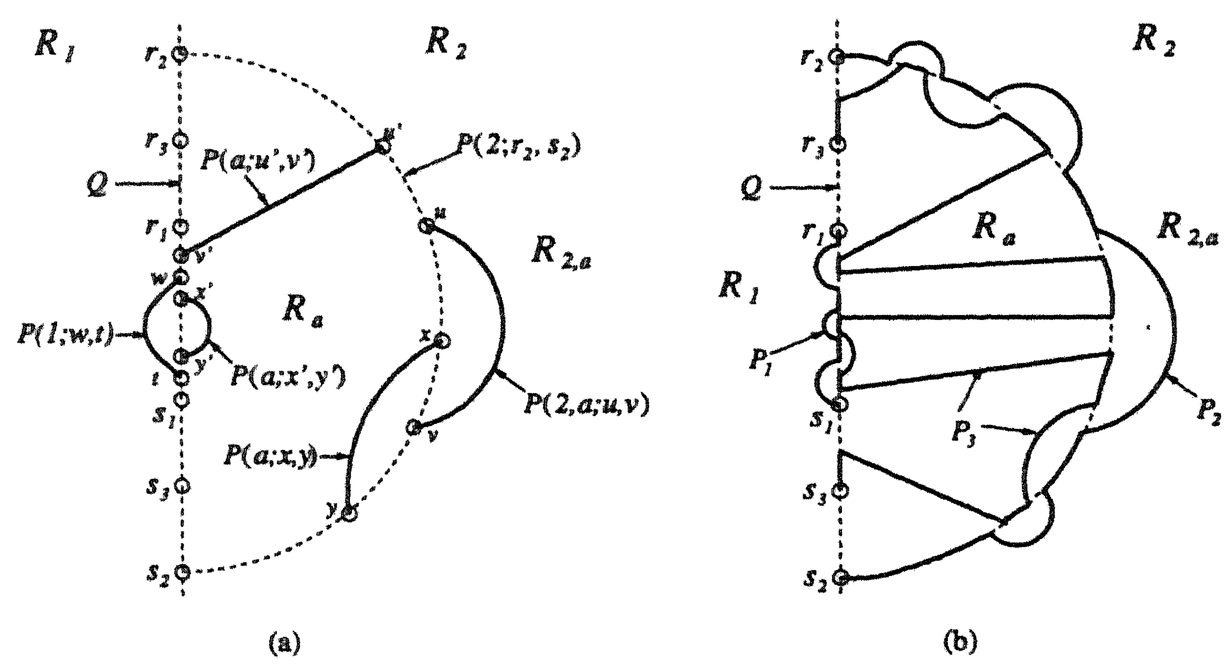


Figure 5.6: Figure (a) illustrates the definitions. Figure (b) shows how we may assume that the paths P_1, P_2, P_3 "look like".

Second step—construction of D_a

In our second step for each arc $a \in A'$ we make a digraph $D_a = (W_a, A_a)$ with lengths $l_1^a, l_2^a, l_3^a : A_a \to \mathbb{Z}_+$. Let us be given $a := ((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$ and suppose a is an arc of type (t).

Let $r_i' :=$ nil if $r_i = r$ and $r_i' := \max\{v \prec r_i \mid v \in V(Q)\}$ otherwise $(i = 1, 2)$. Let $s_i' :=$ nil if $s_i = s$ and $s_i' := \min\{v \succ s_i \mid v \in V(Q)\}$ otherwise. We denote by $G_{3-t,a}$ the graph

$$(R_{3-t} \setminus V(Q(r, r_{3-t}') \cup Q(s_{3-t}', s))) \cup (R_a \setminus V(P(t; r_t, s_t) \setminus \{r, s\}))$$

(where $r_i' =$ nil $\Rightarrow Q(r, r_i') = \emptyset$ and $s_i' =$ nil $\Rightarrow Q(s_i', s) = \emptyset$ $(i = 1, 2)$) and by $G_{t,a}$ the graph

$$(R_t \setminus V(Q(r, r_t') \cup Q(s_t', s))) \setminus (V(Q(r_{3-t}, s_{3-t}) \setminus \{r, s\})).$$

For $i = 1, 2$ if $r_1 \neq r$ then add to $G_{i,a}$ a vertex r_a, zero-length edges $\{r_a, r_i\}$, $\{r_a, r_3\}$ and call the resulting graph by $G_{i,a}$; otherwise define $r_a := r$. For $i = 1, 2$ if $s_1 \neq s$ then add to $G_{i,a}$ a vertex s_a, zero-length edges $\{s_i, s_a\}, \{s_3, s_a\}$ and call the resulting graph by $G_{i,a}$; otherwise define $s_a := s$. Let $P_{t,a}$ be the extension of the path $P(t; r_t, s_t)$ to an $r_a - s_a$-path in $G_{t,a}$ and let $P_{3-t,a}$ be

the extension of the path $Q(r_{3-t}, s_{3-t})$ to an $r_a - s_a$-path in $G_{3-t,a}$. (Observe that both $G_{1,a}$ and $G_{2,a}$ are planar graphs with r_a and s_a adjacent to the unbounded face.)

Make digraphs $D_{i,a} := D_2(G_{i,a}, P_{i,a})$ with length function $l_{i,a}$ and $l_{i+2,a}$ on its arcs ($i = 1, 2$). The length function $l_{3,a}$ (resp. $l_{4,a}$) of $D_{1,a}$ (resp. $D_{2,a}$) corresponds to the path on the right (resp. left) of $P_{1,a}$ (resp. $P_{2,a}$).

The vertex set W_a of D_a consists of quadruples (v_1, v_2, v_3, v_4) so that $v_{3-t}, v_{5-t} \in V(Q(r_{3-t}, s_{3-t})) \cup \{r_3, s_3\}$ and $v_t, v_{t+2} \in V(P(t; r_t, s_t)) \cup \{r_3, s_3\}$. In D_a there is an arc from $v := (v_1, v_2, v_3, v_4)$ to $w := (w_1, w_2, w_3, w_4))$ corresponding to each one of the following conditions satisfied by v, w and $q \in \{1, 2\}$ (We may have more than one arc from v to w corresponding to the same condition. Figure 5.7 illustrates the conditions.):

(i) $(v_{3-q}, v_{5-q}) = (w_{3-q}, w_{5-q})$; $((v'_q, v'_{q+2}), (w'_q, w'_{q+2}))$
is an arc of $D_{q,a}$, where :
$v'_q := r_a$ if $v_q = r_q$ and $v'_q := v_q$ otherwise;
$w'_q := s_a$ if $w_q = s_q$ and $w'_q := w_q$ otherwise;
$v'_{q+2} := r_a$ if $v_{q+2} = r_3$ and $v'_{q+2} := v_{q+2}$ otherwise;
$w'_{q+2} := s_a$ if $w_q = s_3$ and $w'_q := w_q$ otherwise.
In this case for each arc e of $D_{q,a}$ from (v'_q, v'_{q+2}) to (w'_q, w'_{q+2}) we have a corresponding arc d_e of D_a from v to w so that:

$$\begin{aligned} l^a_q(d_e) &:= l_{q,a}(e); \\ l^a_{3-q}(d_e) &:= 0; \\ l^a_3(d_e) &:= l_{q+2,a}(e); \end{aligned}$$

(ii) $(v_3, v_4) = (w_3, w_4)$; $v_{3-q} = w_{3-q}$; $v_{q+2} \prec_a v_q \prec_a w_q$;
For the corresponding arc d define $l^a_{3-q}(d) := l^a_3(d) := 0$ and (5.6)

$$l^a_q(d) := \begin{cases} \text{length}_l(Q(v_q, w_q)) & \text{if } t + q \text{ odd;} \\ \text{length}_l(P(t; v_q, w_q)) & \text{otherwise;} \end{cases}$$

(iii) $(v_1, v_2) = (w_1, w_2)$; $w_{5-q} = v_{5-q}$; $v_{q+2} \prec_a w_{q+2}$; $v_q \prec_a w_{q+2}$.
For the corresponding arc d define $l^a_1(d) := l^a_2(d) := 0$ and
$l^a_3(d) := \text{length}_l(P(a; v_{5-q}, w_{q+2}))$;

(iv) $(v_1, v_2) = (w_1, w_2)$; $v_{5-t} = w_{5-t}$; $v_t \prec_a v_{t+2} \prec_a w_{t+2}$;
For the corresponding arc d define $l^a_1(d) := l^a_2(d) := 0$; and
$l^a_3(d) := \text{length}_l(P(t; v_{t+2}, w_{t+2}))$;

(v) $(v_1, v_2) = (w_1, w_2) = (s_1, s_2)$; $v_{3-q} = w_{3-q} = s_3$; $v_{5-q} \neq s_3$;
For the corresponding arc d define $l^a_1(d) := l^a_2(d) := l^a_3(d) := 0$.

Thus, if $v := (v_1, v_2, v_3, v_4)$ and $w := (w_1, w_2, w_3, w_4)$ fulfill conditions (say) (i) and (iii), then D_a has arcs from v to w corresponding to (i) and to (iii). We say that $d \in A_a$ is an arc of type (t) if d is an arc corresponding to condition

(t) above.

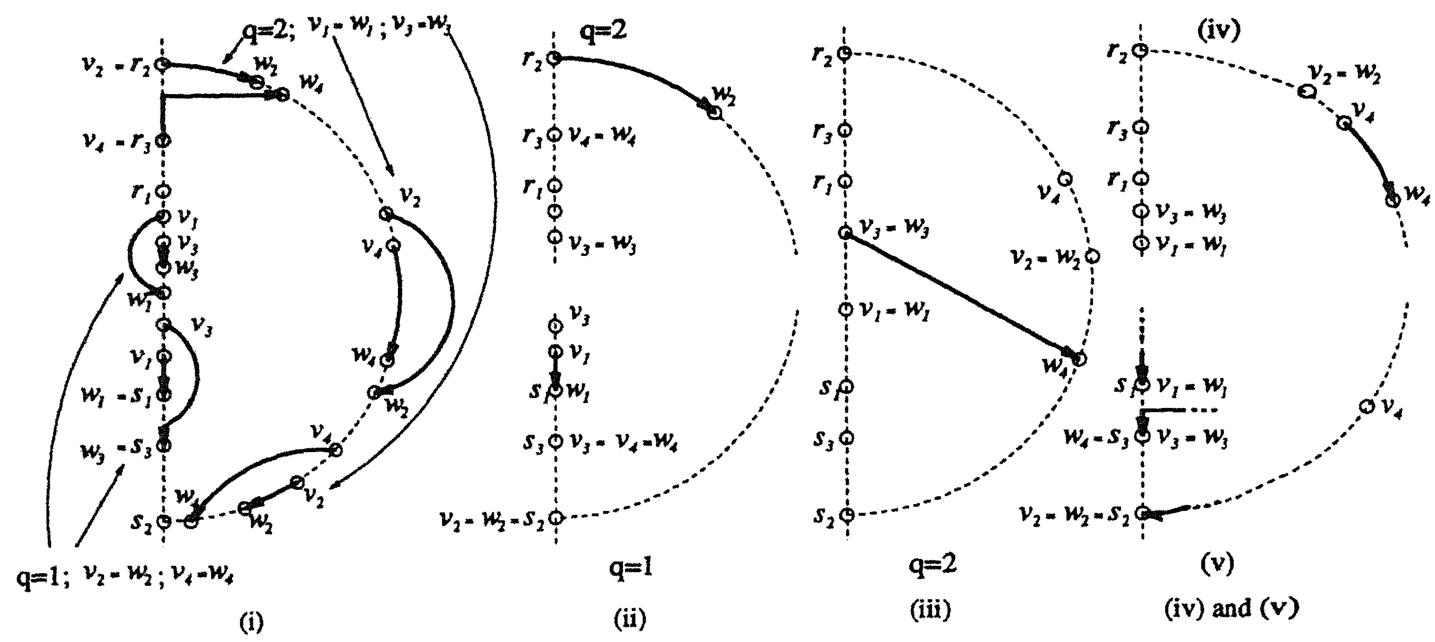


Figure 5.7: $((r_1, r_2), (s_1, s_2)) \in A_2$ is an arc of type (2). Figure (t) illustrates the situation corresponding to a type (t) arc of A_a

Third step—definition of D_3

We are now ready to define the graph $D_3 = (W_3, A_3)$. We make graph D_3 from graph D' by replacing each $a = ((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$ by the graph D_a as follows: delete a from D' and identify the vertices (r_1, r_2, r_3) and (s_1, s_2, s_3) of D' to the vertices (r_1, r_2, r_3, r_3) and (s_1, s_2, s_3, s_3) of D_a, respectively. For each arc d of D_3 and for $i = 1, 2, 3$ we define $l_i(d) := l_i^a(d)$, where $a \in A'$ is so that d is an arc of D_a. This completes the definition of D_3.

Claim 5.1 *The graph D_3 is acyclic and can be constructed in polynomial time.*

Proof. By definition $((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$ implies that (s_1, s_2, s_3) lexicographically greater than (r_1, r_2, r_3) (with respect to $\preceq$). Thus, D' is acyclic. By checking (i)–(v) in the definition of D_a we see that

$$((v_1, v_2, v_3, v_4), (w_1, w_2, w_3, w_4)) \in A_a$$

implies that (w_1, w_2, w_3, w_4) lexicographically greater than (v_1, v_2, v_3, v_4) (with respect to $\preceq_a$). So, D_a is also acyclic for each $a \in A'$. Finally, in the definition of D_3 the direction of the arcs in D_a "agree" to the direction of a. Therefore, D_3 is also acyclic.

By solving $2|A_2| + 2$ all-pairs shortest path problem we can compute:

$$\{P(t, a; u, v) \mid a = ((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A' \text{ of type } (t);$$
$$u, v \in V(P(t; r_t, s_t))\};$$

$$\{P(a;u,v) \mid a = ((r_1,r_2,r_3),(s_1,s_2,s_3)) \in A' \text{ of type } (t);$$
$$u,v \in V(Q(r_t,s_t) \cup P(t;r_t,s_t))\};$$
$$\{P(i;u,v) \mid u,v \in V(Q) \text{ and } i = 1,2\}.$$

Thus, $\{D_a \mid a \in A'\}$ can be computed in polynomial time implying that D_3 can be constructed in polynomial time. ∎

Claim 5.2 *Let P_1, P_2 and P_3 be internally vertex-disjoint paths in G so that:*

(i) P_i is an $r_i - s_i$-path $(i = 1,2,3)$;
(ii) $r_i, s_i \in V(P_i) \cap V(Q)$ $(i = 1,2,3)$;
(iii) no vertex of P_i is an internal vertex of $Q(r_1,r_2)$ or $Q(s_1,s_2)$, $(i = 1,2)$;
(iv) $a := ((r_1,r_2,r_3),(s_1,s_2,s_3)) \in A'$.

Then D_a has an $(r_1,r_2,r_3,r_3) - (s_1,s_2,s_3,s_3)$-path P so that $\text{length}_{l_i}(P) \leq \text{length}_l(P_i)$ $(i = 1,2,3)$.

Proof. We may assume: $a \in A'$ is an arc of type (2); $P_1(r_1,s_1)$ is contained in the graph R_1; $P_3(r_3,s_3)$ is contained in the graph R_a; and $P_2(r_2,s_2)$ is contained in the graph $R_{2,a}$. Moreover, we may assume that for $i = 1,3$ if $u,v \in V(P_i) \cap V(Q(r_2,s_2))$, $u \prec v$ and there exists no $w \in V(P_{4-i})$ so that $u \prec w \prec v$ then $P_i(u,v) = Q(u,v)$. An analogous statement holds if we exchange P_1 by P_2 and Q by $P(2;r_2,s_2)$. (So, we may assume that P_1, P_2 and P_3 "look like" the paths in Figure 5.6(b).)

Let $r' := (r_1,r_2,r_3,r_3)$ and $s' := (s_1,s_2,s_3,s_4)$. Suppose $V(P_3) \cap V(P(2,r_2,s_2)) = \emptyset$. As in the proof of Claim 5.1, there exists in $D_{1,a}$ an $(r_1,r_3) - (s_1,s_3)$-path

$$P' = ((r_1,r_2), a'_1, (v_{1,1},v_{1,2}), \ldots, a'_{d-1}, (v_{d-1,1}, v_{d-1}), a'_d, (s_1,s_3))$$

so that $\text{length}_{l_{1,a}}(P') \leq \text{length}_l(P_i)$, $(i = 1,3)$. Thus,

$$P := ((r_1,r_2,r_3,r_3), a_1, (v_{1,1},r_2,v_{1,2},r_3), \ldots$$
$$\ldots, a_d, (s_1,r_2,s_3,r_3), a_{d+1}, (s_1,s_2,s_3,r_3), a_{d+2}, (s_1,s_2,s_3,s_4)),$$

is an $r' - s'$-path in D_a satisfying the conditions of the claim, where $a_1,\ldots,a_d$ are arcs of type (i), a_{d+1} is an arc of type (ii) and a_{d+2} is an arc of type (v). Analogously, if $V(P_3) \cap V(Q(r_1,s_1)) = \emptyset$ then we are done. So, we may assume P_3 touches both $Q(r_1,s_1)$ and $P(2;r_2,s_2)$.

Let V_1 (resp. V_2) be the set of endpoints of maximal subpaths of $Q(r_1,s_1)$ (resp. $P(2;r_2,s_2)$) whose internal vertices are neither vertices of P_1 (resp. P_2) nor vertices of P_3. We show how traversing simultaneously the paths P_1, P_2, P_3 from r_1, r_2, r_3 to s_1, s_2, s_3, respectively, we can make an $r' - s'$-path P in D_a so that for each $v := (v_1,v_2,v_3,v_4) \in V(P)$ the following holds:

(a) $v_i, v_{i+2} \in V_i \cup \{r_i, s_i, r_3, s_3\}$ $(i = 1, 2)$;
(b) $v_{i+2} \preceq_a v_i \Rightarrow v_{i+2} = \max\{v \preceq_a v_i \mid v \in V(P_3)\}$, $(i = 1, 2)$;
(c) $v_i \preceq_a v_{i+2} \Rightarrow v_i = \max\{v \preceq_a v_{i+2} \mid v \in V(P_i)\}$, $(i = 1, 2)$;
(d) either $v_3 = \max\{v \in V_1 \mid v_4 \in V(P_3(v, s_3))\}$;
or $v_4 = \max\{v \in V_2 \mid v_3 \in V(P_3(v, s_3))\}$;
(e) $\text{length}_{l_i}(P(r, v)) \leq \text{length}_l(P_i(r_i, v_i))$ $(i = 1, 2, 3)$.

Let $v := (v_1, v_2, v_3, v_4)$ and suppose P_v is an $r' - v$-path in D_a satisfying $(a) - (e)$. We show how to chose $w \in W_a$ so that: there exists an $v - w$-arc (i.e., a arc from v to w) $d_{v,w} \in A_a$; w is lexicographically greater than v (with respect to $\preceq_a$); and the $r' - w$-path $P_v + d_{v,w}$ satisfies $(a) - (e)$. (Initially $v = r'$ and $P_{r'}$ clearly satisfies $(a) - (e)$.) Thus repeating this process we eventually make an $r' - s'$-path satisfying the claim.

Suppose $v_3 = s_3$ then (by (c)) $v_1 = s_1$. If $v_2 \neq s_2$ then $w := (s_1, s_2, s_3, v_4)$ is so that there exists a $v - w$-arc $d_{v,w} \in A_a$ of type (iii), otherwise $w := (s_1, s_2, s_3, s_4)$ is so that there exists a $v - w$-arc $d_{v,w} \in A_a$ of type (v). In both cases $P_v + d_{v,w}$ satisfies $(a) - (e)$. Analogously, if $v_4 = s_3$. So, we may assume that $v_3 \prec_a s_3$, $v_4 \prec_a s_3$. Let $v_{i+2}^+ := \min\{v \succ_a v_{i+2} \mid v \in V(P_3) \cap V_i\}$ $(i = 1, 2)$.

Suppose $v_1 = s_1$ and $v_2 = s_2$ and that, say, $v_4 \in V(P_3(v_3, s_3))$. Then $w := (s_1, s_2, v_3, s_4)$ is so that there exists a $v - w$-arc $d_{v,w} \in A_a$ of type (iii) and $P_v + d_{v,w}$ satisfies $(a) - (e)$. Now, suppose that, say, $v_1 = s_1$ and $v_2 \prec_a s_2$ and let $v_2^+ := \min\{v \succ_a v_2 \mid v \in V(P_2) \cap V_2\}$. In this case, since $v_3 \prec_a s_3$, we have $v_3 \prec_a v_1$ and $w := (s_1, v_2^+, v_3, v_4^+)$ is so that there is a $v - w$-arc $d_{v,w} \in A_a$ of type (i) and $P_v + d_{v,w}$ fulfills $(a) - (e)$. So, we may assume $v_1 \prec s_1$. Let $v_1^+ := \min\{v \succ_a v_1 \mid v \in V(P_1) \cap V_1\}$.

Let us assume $v_3 = \max\{v \in V_1 \mid v_4 \in V(P_3(v, s_3))\}$ (the case $v_4 = \max\{v \in V_1 \mid v_3 \in V(P_3(v, s_3))\}$ is similar). The following tables show how to chose w and the type of a $v - w$-arc $d_{v,w} \in A_a$ in the remain possible cases. For all these cases $P_v + d_{v,w}$ fulfills $(a) - (e)$.

case $v_3^+ \in V(P_3(v_4^+, s_3))$

subcase	w	*type* $v - w$-arc
$v_2 \prec_a v_4 \prec_a v_2^+ \prec_a v_4^+$	(v_1, v_2^+, v_3, v_4)	(i)
$v_2 \prec_a v_4 \prec_a v_4^+ \prec_a v_2^+$ and $v_2^+ = \min\{v \in V_2 \mid v \succ_a v_4^+\}$	(v_1, v_2^+, v_3, v_4^+)	(i)
$v_2 \prec_a v_4 \prec_a v_4^+ \prec_a v_2^+$ and $v_2^+ \neq \min\{v \in V_2 \mid v \succ_a v_4^+\}$	(v_1, v_2, v_3, v_4^+)	(iv)
$v_4 \prec_a v_2 \prec_a v_2^+ \prec v_4^+$	(v_1, v_2^+, v_3, v_4^+)	(i)
$v_4 \prec_a v_2 \prec_a v_4^+ \prec_a v_2^+$	(v_1, v_2, v_3, v_4^+)	(i)

case $v_3^+ \notin V(P_3(v_4^+, s_3))$ *subcase*	w	*type* $v - w$-arc
$v_1 \prec_a v_3 \prec_a v_1^+ \prec_a v_3^+$	(v_1^+, v_2, v_3, v_4)	(i)
$v_3 \prec_a v_1 \prec_a v_3^+ \prec_a v_1^+$	(v_1, v_2, v_3^+, v_4)	(iii)
$v_3 \prec_a v_1 \prec_a v_1^+ \prec_a v_3^+$	(v_1^+, v_2, v_3, v_4)	(ii)

∎

It follows from Claim 5.2 that:

Claim 5.3 *Let P_1, P_2 and P_3 be internally vertex-disjoint $r - s$-paths in G. Then D_3 has an $(r_1, r_2, r_3, r_3) - (s_1, s_2, s_3, s_3)$-path P so that $length_{l_i}(P) \leq length_l(P_i)$, $(i = 1, 2, 3)$.* ∎

Claim 5.4 *Let $a := ((r_1, r_2, r_3), (s_1, s_2, s_3)) \in A'$ of type (t) and let P be an $(r_1, r_2, r_3, r_3) - (s_1, s_2, s_3, s_3)$-path in D_a. Then G has internally vertex-disjoint paths P_1, P_2, P_3 so that:*

(i) P_t is an $r_t - s_t$-path in $R_{t,a}$;
P_{3-t} is an $r_{3-t} - s_{3-t}$-path in R_{3-t}; and
P_3 is an $r_3 - s_3$-path in R_a.
(ii) $length_l(P_i) = length_{l_i}(P)$ $(i = 1, 2, 3)$.

Proof. Let $r' := (r_1, r_2, r_3, r_3)$. Assume (say) $t = 2$. By induction on the number d of arcs in the path $P(r', v)$ we show that for each vertex $w := (w_1, w_2, w_3, w_4) \in V(P)$ there are internally vertex-disjoint paths P_{w_1}, P_{w_2} and $P_{w_{3,4}}$ with $w_{3,4} \in \{w_3, w_4\}$, so that:

(a) P_{w_1} is an $r_1 - w_1$-path in R_1;
P_{w_2} is an $r_2 - w_2$-path in $R_{2,a}$;
$P_{w_{3,4}}$ is an $r_3 - w_{3,4}$-path in R_a;
(b) $w_1 = \max\{v \in V(Q) \mid v \in V(P_{v_1})\}$;
$w_2 = \max\{v \in V(P(2; r_2, s_2)) \mid v \in V(P_{v_2})\}$;
$w_3 = \max\{v \in V(Q) \mid v \in V(P_{v_{3,4}})\}$;
$w_4 = \max\{v \in V(P(2; r_2, s_2)) \mid v \in V(P_{v_{3,4}})\}$;
(c) $\text{length}(P_{w_i}) = \text{length}_{l_i}(P(r', w))$ $(i = 1, 2)$;
$\text{length}(P_{w_{3,4}}) = \text{length}_{l_3}(P(r', w))$.

The case $d = 0$ being trivial we assume $d > 0$. Let $v := (v_1, v_2, v_3, v_4) \in W_a$ be the vertex in $P(r', w)$ immediately before w and let $a_d \in A_a$ be the arc from v to w in $P(r', w)$. Let us assume that (say) $v_{3,4} = v_3$. By induction hypothesis there are internally vertex-disjoint paths P_{v_1}, P_{v_2} and P_{v_3} satisfying $(a), (b)$ and (c).

We examine all possible cases depending on the type of arc a_d. We assume that a_d is an arc corresponding to a condition in (5.6) satisfied by v and w for $q = 1$ (the case $q = 2$ is similar).

Firstly, suppose a_d is an arc from v to w corresponding to condition (5.6)(i). Thus, $(v_2, v_4) = (w_2, w_4)$. Suppose $v_1 = v_3 = r$ and $w_1 = w_3 = s$. Hence, a_d is an arc in D_a corresponding to an arc e from (r, r) to (s, s) in $D_{1,a}$. By Claim 5.1 this arc e corresponds to internally vertex-disjoint $r - s$-paths P_1', P_3' in $G_{1,a}$. In this case $P_{w_1} := P_1', P_{w_2} := P_{v_2}$ and $P_{w_3} := P_3'$ are paths satisfying (a), (b) and c. So, we may assume $v_1 \neq r$ (implying $v_3 \neq r$) or $w_1 \neq s$ (implying $w_3 \neq s$) .

For all remain possible cases the paths $P_{w_1}, P_{w_2}, P_{w_3}$ satisfying (a), (b) and (c) are displayed in the table below. For all these cases $P_{w_2} := P_{v_2}$. (Observe that the paths are not necessarily simple.)

type a_d	*case*	P_{w_1}	P_{w_3}
(i)	$v_1 = v_3 = r \prec_a w_3 \prec_a w_1$	$P(1; v_1, w_1)$	$Q(r, w_3)$
	$v_3 = v_1 = r \prec_a w_1 \prec_a w_3$	$Q(r, w_1)$	$P(a; r, w_3)$
	$v_1 \prec_a v_3 \prec_a w_3 = w_1 = s$	$P_{v_1} \cdot P(1; v_1, s)$	$P_{v_3} \cdot Q(v_3, s)$
	$v_3 \prec_a v_1 \prec_a w_1 = w_3 = s$	$P_{v_1} \cdot Q(v_1, s)$	$P_{v_3} \cdot P(a; v_3, s)$
	$v_1 \prec_a v_3 \preceq_a w_3 \prec_a w_1$	$P_{v_1} \cdot P(1; v_1, w_1)$	$P_{v_3} \cdot Q(v_3, w_3)$
	$v_3 \prec_a v_1 \preceq_a w_1 \prec_a w_3$	$P_{v_1} \cdot Q(v_1, w_1)$	$P_{v_3} \cdot P(a; v_3, w_3)$
(ii)		$P_{v_1} \cdot Q(v_1, w_1)$	P_{v_3}
(iii)		P_{v_1}	$P_{v_3} \cdot P(a; v_4, w_3)$
(iv)		P_{v_1}	P_{v_3}
(v)		P_{v_1}	P_{v_3}

∎

Claim 5.5 *Let P be an $(r, r, r, r)-(s, s, s, s)$-path in D_3. Then G has internally vertex-disjoint $r - s$-paths P_1, P_2, P_3 so that $\text{length}_l(P_i) \leq \text{length}_{l_i}(P)$ ($i = 1, 2, 3$).*

Proof. Let $P' := ((r, r, r) = (v_{1,0}, v_{2,0}, v_{3,0}), a_1, \ldots, a_d, (v_{1,d}, v_{2,d}, v_{3,d}) = (s, s, s))$ be the path in D' corresponding to the path P. By Claim 5.4 G has paths $P_{1,1}, P_{2,1}, P_{3,1}, \ldots, P_{1,d}, P_{2,d}, P_{3,d}$ so that

(a) $P_{1,j}, P_{2,j}, P_{3,j}$ are internally vertex-disjoint paths so that:
$P_{t_j,j}$ is an $v_{t_j,j-1} - v_{t_j,j}$-path in R_{t_j,a_j};
$P_{3-t_j,j}$ is an $v_{3-t_j,j-1} - v_{3-t_j,j}$-path in R_{3-t_j,a_j};
$P_{3,j}$ is an $v_{3,j-1} - v_{3,j}$-path in R_{a_1};
where t_j is the type of arc a_j ($j = 1, \ldots, d$);

(b) $\text{length}_l(P_{i,j}) \leq \text{length}_{l_i}(P((v_{1,j-1}, v_{2,j-1}, v_{3,j-1}), (v_{1,j}, v_{2,j}, v_{3,j})))$ ($i = 1, 2, 3; j = 1, \ldots, d$)

We define $P_i' := P_{i,1} \cdot \cdots \cdot P_{i,d}$ ($i = 1, 2, 3$). Thus P_1', P_2', P_3' are $r - s$-paths (not necessarily simple or disjoint) so that $\text{length}_l(P_i') = \text{length}_{l_i}(P)$ ($i = 1, 2, 3$). Now define P_2 as the $r - s$-path in $G[E(P_2')]$ that minimizes the area of the region bounded by P_2 and the $r - s$-path on boundary of G traversed when going from r to s clockwise. By (a) P_1' and P_3' are vertex-disjoint from P_2 and $G[E(P_3')] \setminus V(P_2)$ has an $r - s$-path. Let P_3 be this $r - s$-path so that the area of the region bounded by P_2 and P_3 is minimized. Again, by (a) the graph

$G[E(P_1')] \setminus V(P_3)$ has an $r - s$-path P_1. P_1, P_2 and P_3 as just defined fulfill the requirements of the claim. ∎

Proof of Theorem 5.3 for k=3. By Claim 5.1 D_3 is acyclic and can be constructed in polynomial time. By Claim 5.3 and 5.5, G has internally vertex-disjoint $r - s$-paths P_1, P_2, P_3 so that $\text{length}_l(P_i) \leq b_i$ $(i = 1, 2)$ if and only if D_3 has an $(r, r, r, r) - (s, s, s, s)$-path P so that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1, 2, 3)$. Finally, by Proposition 5.1 an $(r, r, r, r) - (s, s, s, s)$-path P in D_3 so that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1, 2, 3)$ (if any) can be found in $O(|V(G)|^8 L^3)$ time and $O(|V(G)|^4 L^3)$ space, where $L := \max\{b_1, b_2, b_3\}$. ∎

Chapter 6

Train timetabling and bases of circuits

6.1. Introduction

In the present chapter we study the train timetabling problem and we describe how a shortest circuit basis of a graph can be applied in speeding up a backtracking approach to find a *periodic* train timetable. (A basis of the space of circuits of a graph is said a ***shortest circuit basis*** if the sum of the its circuits length is minimum.) The approach described is due to Schrijver [1994] and the train timetable model we consider is due to Schrijver & Steenbeek [1994].

The *train timetabling problem* consists of constructing a train timetable that fulfills a given set of constraints. We are particularly interested in a timetable that repeats itself hourly, i.e. this timetable should be *periodic* with period 60 minutes. Periodicity is a very common constraint in timetabling of public traffic companies; the Nederlandse Spoorwegen (Dutch railways) runs an periodic service on all its routes during most hours of the day (e.g. there is a train leaving Utrecht with final destination The Hague at 7:06h[1], 8:06h, 9:06h, ...).

The set of constraints considered by Schrijver & Steenbeek [1994] are of three types:

- stage-constraints: these constraints determine for each train the duration of each stage of the trip (i.e. duration of the trip between two consecutive stations on the trajectory) and the minimum and maximum stopping time at any station (the difference between the arrival time and departure time);

- market-constraints: these constraints specify train connections and the frequency of trains running on a specific trajectory;

[1]Source: Dutch railways timetable 1994-1995 (*Spoorboekje* 1994-1995).

- infrastructure-constraints: these constraints specify the minimum interval of time which two trains running on the same trajectory must be apart of each other.

The organization of this chapter is as follows. Section 6.2 contains illustrations of the above constraints (in the context of Dutch railways). In Section 6.3 we reformulate the train timetabling problem in terms of the *periodic event scheduling problem* (PESP). Next, in Section 6.4 we discuss a necessary (but not sufficient) condition to feasibility of the PESP. Finally, in Section 6.5 we conclude this chapter by presenting a backtracking approach to the PESP problem and observing that this approach can be carried out more efficiently provided a shortest circuit basis of a graph is given. The model of the periodic scheduling problem we describe is due to Serafini & Ukovich [1989]. and the backtracking approach is due to Schrijver [1994].

6.2. The constraints

A train trip is formed by stages. For instance, the train number 15 of the Dutch railways from Eindhoven to Heerlen is composed of 4 stages:

train #	stage #	trajectory	duration	stopping time
15	00	Eindhoven-Weert	17 min.	[1,3]
15	01	Weert-Roermond	14 min.	[1,3]
15	02	Roermond-Sittard	16 min.	[2,5]
15	03	Sittard-Heerlen	18 min.	—

The train stages of train 15 are numbered 1500, 1501, 1502 and 1503 and the train stages of any train are numbered similarly. We associate with each train stage u a variable t_u representing its departure time.

Stage-constraints

The relation between the departure time of two consecutive train stages is obtained as follows. Consider the train trip of train 15 from Weert to Roermond. The duration of this trip is 14 minutes, the minimum stopping time at Roermond station is 1 minute and maximum stopping time is 3 minutes then the following stage-constraint must hold:

$$t_{1502} - t_{1501} \in [15, 17].$$

This constraint is interpreted modulo 60 meaning that the difference $t_{1502} - t_{1501}$ must be in one of the intervals

$$\ldots, [-105, -103], [-45, -43], [15, 17], [75, 77], [135, 137], \ldots$$

Analogously, $t_{1503} - t_{1504} \in [18, 21]$ (modulo 60) must hold.

Market-constraints

A market-constraint can be used to determine a connection between two trai Consider the train 69 from Heerlen to Roermond and train 8 from Maastric to Haarlem. These two trains stop at Sittard were a connection must oc from train 69 to train 8 with minimum transfer time of 2 minutes and maximu transfer time of 6 minutes (here the transfer time is the difference between t arrival time at Sittard of train 69 and departure time from Sittard of tra 8). The stage of train 8 leaving Sittard is stage 01 and the stage of train arriving at Sittard is stage 00. The running time of train stage 6900 is minutes. Thus a timetable satisfying the following market-constraint realiz the desired connection

$$t_{801} - t_{6900} \in [24, 28].$$

A constraint enforcing the frequency of trains running on a specific tr jectory is also a market-constraint. As an example consider train 7 from Haa lem to Eindhoven and train 27 from Haarlem to Maastricht. These trains ha the whole trajectory (of about 100 minutes) from Haarlem to Eindhoven common. Hence, it is convenient that the hourly departure time of train 7 30 minutes apart from the departure time of train 27. In order to fulfill th requirement we demand that

$$t_{2700} - t_{700} = 30, \quad t_{2701} - t_{701} = 30, \ \ldots$$

Infrastructure-constraints

Infrastructure-constraints specify the minimal interval of time which two trai running on the same rail must be apart of each other. This type of constrai also takes into account crossings. (For our purposes a crossing can be regarde as a stations where the minimum and maximum stopping time of any train 0 minutes.) For instance the constraint

$$t_{5320} - t_{19735} \in [3, 57]$$

implies that the departure time of train stage 5320 and train stage 19735 (bot from Lelystad Centrum to Almere Buiten) must be at least 3 minutes apart.

Another example of an infrastructure constraint is

$$t_{22450} - t_{12400} \in [53, 101].$$

This constraint is intended to guarantee that the departure time of train stag 22450 from 's-Hertogenbosch to Nijmegen and train stage 12400 from Nijmege to 's-Hertogenbosch is not in the interval [42,52]. This is necessary due th existence on the trajectory of a bridge with only one rail.

6.3. The model—the periodic event scheduling problem

In the previous section the train timetabling problem was reduced to the problem of finding a solution to a set C of constraints where each constraint is of the form

$$t_{i_j} - t_{i'_j} \in [d_j^-, d_j^+] \quad \text{(modulo 60)}. \tag{6.1}$$

Associated with this set of constraints we define a directed graph $D = (V, A)$, where V is the set of variables in C and for each constraint in C of the form (6.1) there exists in D an arc from variable $t_{i'_j}$ to variable t_{i_j}. Thus, the train timetabling problem can be easily seen to be equivalent to the following *periodic event scheduling problem*:

given : – a connected digraph $D = (V, A)$;
– a positive integer T;
– functions $d^-, d^+ : A \to \mathbb{Z}$;

find : – functions $\pi : V \to \mathbb{Z}$ and $z : A \to \mathbb{Z}$ so that for each arc $a = (u, v)$ of D the following holds: (6.2)

$$d^-(a) \leq \pi(v) - \pi(u) + z(a)T \leq d^+(a).$$

The interpretation of the problem is as follows. Each vertex of D corresponds to a *periodic event*, i.e. an event that must occur at intervals of exactly T units of time. T is called the *period* of the event. As an example of periodic event consider the intercity train from Zandvoort aan Zee to Maastricht whose departure time from (say) Haarlem occurs at 8:42h, 9:42h, 10:42h,.... So, the departure time of this train from Haarlem constitutes an event of period $T = 60$ minutes. (In the train timetabling problem T is typically 60 minutes.)

We associated a *span interval* $[d^-(a), d^+(a)]$ with each arc $a = (u, v)$ of D. The integers $d^-(a)$ and $d^+(a)$ give the minimum and maximum time interval, respectively, between the occurrence of event u and event v. The integer $d^-(a)$ can be, for example, used to model the duration of the event u and the safety gap between the occurrence of event v and u. Suppose events u and v correspond to the departure time of two trains t_u and t_v, respectively, which use the same trajectory. It is advisable for safety precautions that train t_v departs at least (say) 3 minutes before train t_u, in this case $d^-(a) = 3$ (this requirement corresponds to an infrastructure-constraint). On the other hand, people willing to transfer from train t_u to train t_v will not be very happy if they have to wait too long till the departure time of train t_v; $d^+(a)$ can be used to model the maximum amount of time this people will possibly have to wait, including the time that the train stays at the platform. (Observe that it does not make any difference to the problem whether or not we consider the span $[d^-(a), d^+(a)]$ or the span $[d^-(a) + pT, d^+(a) + pT]$ for some $p \in \mathbb{Z}$.)

The function π is called a *potential* and the corresponding function $\delta : A \to \mathbb{Z}$ so that $\delta(a) := \pi(v) - \pi(u)$ for each arc $a = (u, v)$ of D is called a

potential difference (equivalently, we call a potential difference any function $\delta : A \to \mathbb{Z}$ satisfying Kirchhoff's voltage law). Problem (6.2) can be reformulated as the problem of finding a potential difference δ and a function $z : A \to \mathbb{Z}$ so that:

$$d^-(a) \leq \delta(a) + z(a)T \leq d^+(a) \quad \text{for each arc } a \text{ of } D.$$

The potential π gives the scheduling of the event set V. For each $v \in V$ the event v will occur at instants $\pi(u), \pi(u) \pm T, \pi(u) \pm 2T, \ldots$ (We identify the time with the real line.) If π is a solution to (6.2) then π_p is also a solution to (6.2) for all $p \in \mathbb{Z}$, where $\pi_p(v) := \pi(v) + pT$ for each $v \in V$. Thus, if $v \in V$ is an event corresponding to the departure time of train t from station s and π is a solution to (6.2) so that $\pi(v) = -678$, then this should be interpreted as: train t departs from station s at $\ldots$, 8:42, 9:42, 10:42, $\ldots$ ($-678+12 \cdot 60 = 42 \equiv 678$ modulo 60).

The presence of the function z in problem (6.2) is due to a technical reason. We could as well delete z from the enunciate and consider instead d^-, d^+ and π (or δ) as functions having the residual class of $\mathbb{Z}$ modulo $60\mathbb{Z}$ ($\mathbb{Z}/60\mathbb{Z}$) as counter-domain.

Problem (6.2) is $\mathcal{NP}$-complete in the strong sense (cf. Daucha, Modrow & Neumann [1985]). However, for any fixed z problem (6.2) is known as the *feasible differential problem* and can be solved in $O(nm)$ time (see, e.g., Rockafellar [1984: Ch. 6]). In the sequel we present a backtracking approach that solves problem (6.2) by solving (exponential) many feasible differential problems.

6.4. A necessary condition to feasibility

As observed by Serafini & Ukovich [1989] the feasibility of problem (6.2) is not affected by fixing arbitrarily the value of $z(a)$ for all arcs a of a spanning tree $H = (V, A_0)$ of D. Indeed, let $b \in A_0$, π and z be a solution to problem (6.2) and V^+ be the vertex set of the connected component of $H - b$ containing the head of b. For $p \in \mathbb{Z}$ consider the following functions $\pi_{b,p} : V \to \mathbb{Z}$ and $z_{b,p} : A \to \mathbb{Z}$:

$$\pi_{b,p}(v) := \begin{cases} \pi(v) + (z(b) - p)T & \text{if } v \in V^+; \\ \pi(v) & \text{otherwise;} \end{cases}$$

$$z_{b,p}(a) := \begin{cases} z(a) - z(b) + p & \text{if } a \in \delta^+(V^+); \\ z(a) + z(b) - p & \text{if } a \in \delta^-(V^+); \\ z(a) & \text{otherwise.} \end{cases}$$

One easily checks that $\pi_{b,p}$ and $z_{b,p}$ form a solution to (6.2) so that $z_{b,p}(b) = p$. Repeating this we can make $z(a) = 0$ for each $a \in A_0$. Thus, in addition without loss of feasibility we may assume that z in (6.2) satisfies:

$$z(a) = 0 \quad \text{for all } a \in A_0.$$

To each (undirected) path

$$P = (v_0, a_1, v_1, a_2, \ldots, v_{t-1}, a_t, v_t),$$

of the digraph D we associate a partition (P^+, P^-) of its arc set where

$$\begin{aligned} P^+ &:= \{a_i \mid i \in \{1, \ldots, t\} \text{ and } a_i = (v_{i-1}, v_i)\} \text{ and} \\ P^- &:= \{a_i \mid i \in \{1, \ldots, t\} \text{ and } a_i = (v_i, v_{i-1})\}. \end{aligned}$$

For each path P of D and function $z : A \to \mathbb{Z}$ we define

$$z(P) := \sum_{a \in P^+} z(a) - \sum_{a \in P^-} z(a)$$

We define the *lower span* $l(C)$ and the *upper span* $u(C)$ of a circuit C of D as follows:

$$\begin{aligned} l(C) &:= \sum_{a \in C^+} d^-(a) - \sum_{a \in C^-} d^+(a); \\ u(C) &:= \sum_{a \in C^+} d^+(a) - \sum_{a \in C^-} d^-(a). \end{aligned}$$

(Thus, l and u are functions on the circuit set of D.)

Let $\mathcal{C}$, $\mathcal{B}$ and $H = (V, A_0)$ be the circuit set, a basis of circuits and a spanning tree of D, respectively. (Recall that the circuit space of a digraph $D = (V, A)$ is the linear subspace of $\mathbb{R}^A$ generated by $\{0, \pm 1\}$-vectors corresponding to circuits of D.) The *Feasible Differential Theorem* states that the feasible differential problem has a solution if and only if $l(C) \leq 0 \leq u(C)$ for all $C \in \mathcal{C}$ (cf. Minty [1960]—see also Rockafellar [1984: Ch. 6]). A similar trivial necessary condition to the feasibility of the periodic event scheduling problem (6.2) is:

there exists $z : A \to \mathbb{Z}$ *so that:*

$$\begin{aligned} &(i)\ z(a) = 0 \textit{ for all } a \in A_0; \textit{ and} \\ &(ii)\ l(C) \leq z(C)T \leq u(C) \textit{ for all } C \in \mathcal{B}. \end{aligned} \tag{6.3}$$

Note that condition (6.3) is not sufficient—see Figure 6.1(a). It follows from the Feasible Differential Theorem that (6.3) would become a sufficient condition if we demand that

$$l(C) \leq z(C)T \leq u(C) \quad \text{for all } C \in \mathcal{C} \tag{6.4}$$

(instead of for all $C \in \mathcal{B}$). Indeed, if $z : A \to \mathbb{Z}$ is a function satisfying (6.4) then solving the periodic event scheduling problem (6.2) reduces to find a function $\pi : V \to \mathbb{Z}$ so that for each arc $a = (u, v)$ of D the following holds:

$$d^-(a) - z(a)T \leq \pi(v) - \pi(u) \leq d^+(a) - z(a)T. \tag{6.5}$$

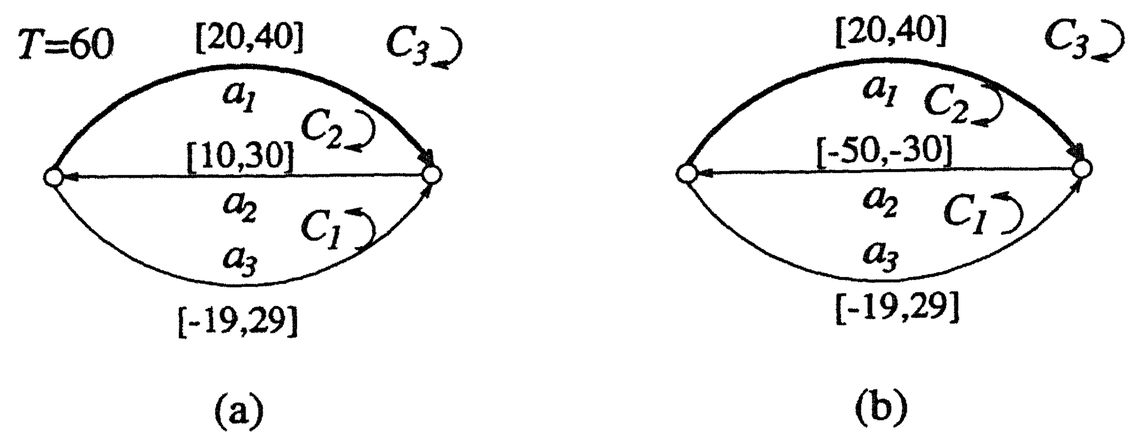


Figure 6.1: (a) An example of an infeasible periodic event scheduling problem that satisfies condition (6.3) by taking $A_0 = \{a_1\}$, $\mathcal{B} = \{C_1, C_2\}$, $z(a_1) = 0$, $z(a_2) = 1$ and $z(a_3) = -1$. (b) A feasible differential problem with no solution (here we are taking $z := 0$); the upper span of the circuit C_1 is negative.

From (6.4) it follows that $d^-(C) - z(C)T \leq 0 \leq d^+(C) - z(C)T$ for all $C \in \mathcal{C}$. This implies, by the Feasible Differential Theorem, that there exists a function π fulfilling (6.5).

6.5. A backtracking algorithm

Let us be given a spanning tree $H = (V, A_0)$ and a basis of circuits $\mathcal{B}$ of D. The algorithm consists of enumerating all functions $z : A \to \mathbb{Z}$ satisfying conditions (6.3) and solving for each of these functions the feasible differential problem obtained from problem (6.2) by fixing z. More specifically, given $f : \mathcal{B} \to \mathbb{Z}$ there exists a unique $z_f : A \to \mathbb{Z}$ so that:

$$\begin{array}{rcll} z_f(a) & = & 0 & \text{for all } a \in A_0; \\ z_f(C) & = & f(C) & \text{for all } C \in \mathcal{B}. \end{array} \tag{6.6}$$

The uniqueness and integrality of z_f are due to the fact that the $m \times m$-matrix of the above system of equalities is a unimodular matrix (i.e. its determinant is ± 1), as can be easily verified (see Schrijver [1986: Ch. 19]). As an illustration we show in (6.7) the system associated with the problem in Figure 6.1(a) by taking $A_0 = \{a_1\}$, $\mathcal{B} = \{C_2, C_3\}$, $f(C_2) = 1$ and $f(C_3) = 0$.

$$\begin{array}{lcl} z_f(a_1) & = & 0 \\ z_f(a_1) + z_f(a_2) & = & 1 \\ z_f(a_1) - z_f(a_3) & = & 0 \end{array} \tag{6.7}$$

The solution of the above system is $z_f(a_1) = z_f(a_3) = 0$ and $z_f(a_2) = 1$. The corresponding feasible differential problem obtained by taking $z := z_f$, shown in Figure 6.1(b), has no solution as the upper span of the circuit C_1 is negative. Hence, in order to enumerate all z satisfying (6.3) we simply enumerate all $f : \mathcal{B} \to \mathbb{Z}$ satisfying $l(C) \leq f(C)T \leq u(C)$ for all $C \in \mathcal{B}$ and solve the corresponding system (6.6) for each one of them.

Let $G_D = (V, A')$ be the underlying graph of D. For each arc a and circuit C of D let a' and C' denote the corresponding edge and underlying circuit of G_D, respectively. Thus, if $\mathcal{B} = \{C_1, \ldots, C_{\nu(D)}\}$ is a circuit basis of D then $\mathcal{B}' := \{C'_1, \ldots, C'_{\nu(D)}\}$ is a circuit basis of G_D, where $\nu(D)$ is the dimension of the circuit space of D. (Recall that the circuit space of an undirected graph with edge set E is a linear subspace of $\mathrm{GF}(2)^E$ and note that the dimension of the circuit space of G_D is equal to $\nu(D)$.)

Let $d : A' \to \mathbb{Z}$ be so that $d(a') := d^+(a) - d^-(a)$. The number of possible values we are allowed to attribute to $f(C)$ in (6.6), $C \in \mathcal{B}$, is bounded above by $\lceil d(C')/T \rceil$ where $d(C') = \sum_{a' \in C'} d(a')$. Therefore, the total number of functions the algorithm possibly need to enumerate (or, in other words, the number of "leaves" of our "decision tree") is bounded above by

$$\prod_{C' \in \mathcal{B}'} \lceil d(C')/T \rceil \tag{6.8}$$

The resulting algorithm runs in time $O((\prod_{C \in \mathcal{B}} d(C)) \cdot nm/T^{\nu(D)})$.

In minimizing (6.8) is where a shortest circuit basis come into play. A circuit basis $\mathcal{B}'$ is a shortest circuit basis of G_D (with respect to the length function d) provided that

$$\sum_{C' \in \mathcal{B}'} \sum_{a' \in C'} d(a') \text{ is minimum.}$$

(Thus, a shortest circuit basis of an undirected graph is a circuit basis whose the sum of the circuits length is minimum.) In Chapter 7 we prove that by taking $\mathcal{B}'$ as a shortest circuit basis of G_D one does minimize (6.8) (cf. Corollary 7.1). Chapter 7 contains also the description of polynomial-time algorithms for finding a shortest circuit basis of an undirected graph. The train timetabling problem and the above backtracking approach was our main motivation to study the "shortest circuit basis problem".

Remark 6.1. It is possible to add a very trivial "bounding approach" to this backtracking algorithm. Let $\{C_1, \ldots, C_{\nu(D)}\}$ be the circuit basis $\mathcal{B}$. We may rewrite system (6.6) as follows:

$$\begin{array}{rcll} z_f(a) & = & 0 & \text{for all arc } a \in A_0; \\ z_f(a) & = & \sum_{i=1}^{\nu(D)} \lambda_{a,i} f(C_i) & \text{for all arc } a \in A \setminus A_0, \end{array} \tag{6.9}$$

where $\lambda_{a,i} \in \{0, \pm 1\}$ ($a \in A \setminus A_0$; $i = 1, \ldots, \nu(D)$). Suppose we have fixed the values of $f(C_1), \ldots, f(C_k)$, $k < \nu(D)$, and in doing this the arcs for which the value of z_f is already defined by system (6.9) gives rise to a circuit with negative upper span or positive lower span in the corresponding feasible differential problem. In this case, we know that there exists no possible way of choosing values to $f(C_{k+1}), \ldots, f(C_{\nu(D)})$ in order to obtain a z_f that leads to a solution to problem (6.2). Thus we can abandon the present branch of the decision tree and go to another one. One can also use the system (6.9) in order to choose a

order of the circuits in $\mathcal{B}$ in such a way that by fixing the values of f in this order we get "many" defined values of z_f; this might help to detect in advance a circuit with negative upper span or positive lower span and avoid the search in a branch of the decision tree.

Remark 6.2. Consider the following trivial necessary condition to the feasibility of the periodic event scheduling problem:

for each circuit C of D it holds that: (6.10)

$$\lceil l(C)/T \rceil \leq \lfloor u(C)/T \rfloor.$$

We do not know how to test condition (6.10) in polynomial time. However, the following stronger condition can be tested in polynomial time for each fixed T:

for each $v, w \in V$ there exists a nonempty set $J_{v,w}$ of $\mathbb{Z}/T\mathbb{Z}$ so that:

(6.11)

(*i*) $J_{v,v} = \{0\}$ *for each* $v \in V$;

(*ii*) $J_{u,v} = -J_{v,u}$ *(i.e.* $i \in J_{u,v}$ *if and only if* $-i \in J_{v,u}$*) for each* $u, v \in V$;

(*iii*) $J_{u,v} \subseteq J_{u,w} + J_{w,v}$ *(i.e.* $J_{u,v} \subseteq \{i + j \mid i \in J_{u,w},\ j \in J_{w,v}\}$*) for each* $u, v, w \in V$;

(*iv*) $J_{u,v} \subseteq [d^-(a), d^+(a)]$ *for each arc* $a = (u, v)$ *of* D.

To see that (6.11) is indeed a necessary condition consider a solution π and z of the periodic event scheduling problem (6.2) and define

$$J_{u,v} \quad := \quad \{(\pi(v) - \pi(u)) \text{ modulo } T\} \quad \text{for each } u, v \in V.$$

One can easily verify that these sets $J_{u,v}$ as defined satisfy (6.11). To see that condition (6.11) is stronger than (6.10) consider now sets $(J_{u,v})_{u,v \in V}$ satisfying (6.11) and let

$$C \quad := \quad (v_0, a_1, v_1, a_2, \ldots, v_{t-1}, a_t, v_t = v_0),$$

be an arbitrary circuit of D. By (6.11) (*i*) and (*iii*) we have that

$$\{0\} = J_{v_0,v_0} \subseteq J_{v_0,v_1} + J_{v_1,v_2} + \cdots + J_{v_{t-1},v_0},$$

that is there exist integers $\delta_i \in J_{v_{i-1},v_i}$ $(i = 1, \ldots, t)$ so that

$$kT \quad = \quad \delta_1 + \delta_2 + \cdots + \delta_t$$

for some integer k. Finally, from (6.11) (*iv*) it follows that

$$l(C) \leq kT \leq u(C).$$

Condition (6.11) can be tested in $O(n^2T)$ time by the classic Bellman-Ford-Moore's method (i.e. solution by successive approximations). Initially we set

$$J_{u,v} := \begin{cases} \{0\} & \text{if } u = v; \\ \{d^-(a), \ldots, d^+(a)\} & \text{if } u \neq v \text{ and } a = (u,v) \text{ is an arc of } D; \\ \{-d^+(a), \ldots, -d^-(a)\} & \text{if } u \neq v \text{ and } a = (u,v) \text{ is an arc of } D; \\ \{0, \ldots, T-1\} & \text{otherwise,} \end{cases}$$

and proceed setting

$$J_{u,v} := J_{u,v} \cap (J_{u,w} + J_{w,v}) \quad \text{and} \quad J_{v,u} := -J_{u,v}$$

for each $J_{u,v}, J_{u,w}, J_{w,v}$ not satisfying (6.11)(*iii*). This process terminates ("stabilizes") after at most $O(n^2T)$ iterations.

Remark 6.3. As observed by Horton [1987], in most applications (at least in all applications we are aware of) a shortest circuit basis is not wanted for its own sake, but to be used as input for a latter algorithm in order to reduce the amount of work that has to be done by this latter algorithm. The application we have just described and the one in Section 7.6 are both of this type.

Chapter 7

Shortest bases of circuits

7.1. Introduction

In the present chapter we consider the following *shortest circuit basis problem:*

given : – an undirected connected graph $G = (V, E)$;
– a length function $l : E \to \mathbb{R}^+$;

find : – a basis $\mathcal{B}$ of the circuit space of G so that (7.1)

$$\sum_{C \in \mathcal{B}} \sum_{e \in E(C)} l(e) \text{ is minimum.}$$

This problem can be solved by the greedy algorithm, but that requires listing all circuits of G in advance.

Circuit bases have been used to study the cyclic structure of a graph since the time of Gustav Kirchhoff (see Section 7.5). There are several algorithms that use a circuit basis of a graph (cf. Sysło [1981] and the references in it). Some of these algorithms can be sped up if a shortest circuit basis is given (see, e.g., Chapter 6 of this thesis). This is already a reason for one's interest in an efficient algorithm to find a shortest circuit basis.

Applications and history of the shortest circuit basis problem are surveyed, for instance, in Deo, Prabhu & Krishnamoorthy [1982] and Horton [1987]. Let us repeat here a few historical considerations. The oldest reference to the shortest circuit basis problem we were able to find is Stepanets [1964] (I am indebted to Paulo Feofiloff for kindly translating this paper to me). This problem was later considered by Zykov [1969]. Hubicka & Sysło [1976] have shown that the Stepanets-Zykov's algorithm fails in finding a shortest circuit basis; they proposed a new algorithm to find a *short* circuit basis and conjectured that their method actually finds a shortest circuit basis. Kolasinska [1980] exhibited a graph for which Hubicka-Sysło's algorithm does not succeed in finding a shortest circuit basis. All of theses algorithms dealt with the special case when

all edges have length one. The shortest circuit basis problem was conjectured to be $\mathcal{NP}$-complete by Kaveh [1989]—this is clearly the case when negative edges length are allowed. Finally, Horton [1987] showed that the shortest circuit basis problem (7.1) can be solved in polynomial time for graphs with nonnegative edge lengths.

Closely related is the problem of finding a shortest *fundamental circuit basis*, i.e.: *given*: a undirected connected graph G; *find*: a spanning tree T of G so that the fundamental circuit basis $\{C(T,e) \mid e \in E(G) \setminus E(T)\}$ is as short as possible. This problem was considered by Hubicka & Sysło [1976] and shown to be $\mathcal{NP}$-complete by Deo, Prabhu & Krishnamoorthy [1982].

Horton's algorithm finds a shortest circuit basis of a graph with n vertices and m edges in $O(m^3 n)$ time. In Section 7.2 we describe an $O(m^3 + mn^2 \log n)$-time algorithm for the shortest circuit basis problem. In Section 7.3 we discuss the close relation between circuit bases and odd circuits in signed graphs. Next, in Section 7.4, we describe Horton's polynomial-time algorithm. In Section 7.5 we study the relation between the shortest circuit basis problem and polyhedral theory. In Section 7.6 we describe a well-known application of shortest circuit bases to the cycle analysis of electrical networks. Finally, in Section 7.7 we give a few last remarks on the problem. Before proceeding we present here the notation we shall need in the sequel.

Some additional notation and terminology

Let $G = (V, E)$ be a undirected connected graph and $\mathcal{C}$ be the set of cycles of G. We denote by $n(G)$, $m(G)$ and $\nu(G)$ (or, simply, n, m, and ν) the number of vertices, the number of edges and the dimension of the circuit space of G, respectively. As is well-known $\nu = m - n + 1$.

We do not make any distinction between a cycle (or path) and its edge set. Thus, if P, Q and T are paths or cycles of G we write $P = Q \,\Delta\, T$ meaning that $E(P) = E(Q) \,\Delta\, E(T)$.

For $\mathcal{F} \subseteq \mathcal{C}$ let $\langle \mathcal{F} \rangle := \{C_1 \,\Delta\, \cdots \,\Delta\, C_t \mid C_i \in \mathcal{F}\ (i = 1, \ldots, t)\}$ (i.e., $\langle \mathcal{F} \rangle$ is the linear subspace of $\mathrm{GF}(2)^{E(G)}$ generated by the circuits in $\mathcal{F}$). Thus, in particular, if $\mathcal{B}$ is a circuit basis of G then $\langle \mathcal{B} \rangle = \mathcal{C}$. We say that circuits $C_1, \ldots, C_k$ (or signs $\Sigma_1, \ldots, \Sigma_k$) of G are *independent* if their incidence vectors are linear independent in $\mathrm{GF}(2)^{E(G)}$.

7.2. A polynomial-time algorithm

The first polynomial-time algorithm for the shortest circuit basis problem is due to Horton [1987]. The running time of Horton's algorithm is $O(m^3 n)$. In this section we show that problem (7.1) can be solved in time $O(\nu^3)$ plus the time of $\nu(G)$ applications of an algorithm for the following *shortest odd circuit*

problem:

given : – a signed graph (G, Σ);
– a length function $l : E \to \mathbb{R}_+$;

find : – an odd circuit C in (G, Σ) so that (7.2)

$$\sum_{e \in E(C)} l(e) \text{ is minimum.}$$

(Recall that an odd circuit in (G, Σ) is a circuit of G containing an odd number of edges in Σ.) The resulting algorithm can be seen to find a shortest circuit basis in $O(m^3 + mn^2 \log n)$ time.

The algorithm

Let us be given an undirected connected graph $G = (V, E)$ and a nonnegative length function $l : E \to \mathbb{R}_+$. We find a shortest circuit basis of G as follows:

Step 1. Choose a spanning tree T of G. Let $\{e_1, \ldots, e_\nu\}$ be the edges not in T (i.e. $\{e_1, \ldots, e_\nu\}$ is a spanning cotree of G.)
Define $\Sigma_{1,i} := \{e_i\}$ $(i = 1, \ldots, \nu)$.

Step 2. For $k := 1, \ldots, \nu$ iterate the following:

Find a shortest odd circuit C_k in $(G, \Sigma_{k,k})$ (with respect to the length function l);

Define for $i = k+1, \ldots, \nu$:

$$\Sigma_{k+1,i} := \begin{cases} \Sigma_{k,i} \,\Delta\, \Sigma_{k,k} & \text{if } C_k \text{ is an odd circuit in } (G, \Sigma_{k,i}) \\ \Sigma_{k,i} & \text{otherwise} \end{cases}$$

The algorithm returns $\{C_1, \ldots, C_\nu\}$.

Correctness of the algorithm

The correctness is based on a few easy-to-prove lemmas. First we note that $\Sigma_{k,k}$ is not in the cut space of G as $\Sigma_{k,k} \neq \emptyset$ $(e_k \in \Sigma_{k,k})$ and $\Sigma_{k,k} \subseteq \{e_1, \ldots, e_\nu\}$ $(k = 1, \ldots, \nu)$. Therefore, $(G, \Sigma_{k,k})$ has an odd circuit for $k = 1, \ldots, \nu$ and Step 2 indeed makes sense in each iteration.

Lemma 7.1 *Let C be a circuit of G so that C is an even (odd) circuit in (G, Σ) and in (G, Σ'). Then C is an even circuit in $(G, \Sigma \,\Delta\, \Sigma')$.* ∎

Lemma 7.2 *Let $C_1, \ldots, C_k$ be independent circuits and let Σ be a sign of G so that C_i is an even circuit in (G, Σ) $(i = 1, \ldots, k)$. Then $C_1, \ldots, C_k, C$ are independent circuits for any odd circuit C in (G, Σ).* ∎

Lemma 7.3 *Let C_k be the odd circuit in $(G, \Sigma_{k,k})$ found during iteration k of Step 2 ($k = 1, \ldots, \nu - 1$). Then C_k is an even circuit in $(G, \Sigma_{p,i})$ ($k+1 \leq p \leq i \leq \nu$).*

Proof. By Lemma 7.1 and the definition of $\Sigma_{k+1,i}$ ($i = k+1, \ldots, \nu$) it follows that C_k is an even circuit in $(G, \Sigma_{k+1,i})$. For $p > k+1$ the lemma follows by induction from Lemma 7.1 since each $\Sigma_{p,i}$ is either $\Sigma_{p-1,i}$ or $\Sigma_{p-1,i} \Delta \Sigma_{p-1,p-1}$. ∎

The following theorem establishes the correctness of the algorithm.

Theorem 7.1 *The set $\mathcal{B} := \{C_1, \ldots, C_\nu\}$ returned by the algorithm is a shortest circuit basis of G.*

Proof. From Lemma 7.2 and Lemma 7.3 it follows that $C_k \notin \langle\{C_1, \ldots, C_{k-1}\}\rangle$ ($k = 2, \ldots, \nu$). Thus $\mathcal{B}$ is a circuit basis.

Suppose by contradiction that $\mathcal{B}$ is not a shortest circuit basis and let $\mathcal{B}'$ be a shortest circuit basis of G so that

$$k := \min\{i \in \{1, \ldots, \nu\} \mid C_i \in \mathcal{B} \setminus \mathcal{B}'\} \text{ is as large as possible.} \tag{7.3}$$

We have that

$$C_k \;=\; C'_1 \Delta C'_2 \Delta \cdots \Delta C'_p,$$

for distinct circuits $C'_1, \ldots, C'_p$ of $\mathcal{B}'$. Thus, there must exist a $t \in \{1, \ldots, p\}$ so that C'_t is an odd circuit in $(G, \Sigma_{k,k})$ implying that $l(C_k) \leq l(C'_t)$. Hence, $(\mathcal{B}' \setminus \{C'_t\}) \cup \{C_k\}$ is a shortest circuit basis of G for which the minimum in (7.3) is larger than k, a contradiction. (Observe that $C'_t \notin \{C_1, \ldots, C_{k-1}\}$ as, by Lemma 7.3, C_i is an even circuit in $(G, \Sigma_{k,k})$ ($i = 1, \ldots, k-1$).) ∎

Complexity of the algorithm

The time-dominating step of the algorithm described is Step 2 (Step 1 can be carried out in linear time.) The following well-known construction reduces the shortest odd circuit problem to n single-source shortest path problems. Split each vertex v into vertices v^+ and v^-. For each edge $\{u, v\} \in \Sigma$ make new edges $\{u^+, v^-\}$ and $\{u^-, v^+\}$ both of length $l(\{u, v\})$. For each edge $\{u, v\} \notin \Sigma$ make new edges $\{u^+, v^+\}$ and $\{u^-, v^-\}$, both of length $l(\{u, v\})$. Now, for each vertex v of G we compute a shortest $v^+ - v^-$-path in the new graph. The shortest among these paths will give us a shortest odd circuit in (G, Σ). This algorithm runs in time $O(n(m + n \log n))$ (see Preliminaries on the shortest path problem). (The shortest odd circuit problem can also be reduced to m minimum weighted perfect matching problems (cf. Grötschel & Pulleyblank [1981]; see also Grötschel, Lovász & Schrijver [1988: Ch. 8]).)

The time complexity of updating the signs $\Sigma_{k,k}, \ldots, \Sigma_{k,\nu}$ at iteration $k-1$ of Step 2 is $O(k(\nu - k))$. So, the total time required by the algorithm in

updating signs is

$$O(\sum_{k=1}^{\nu-1} k(\nu-k)) = O(\nu^3).$$

Therefore, the overall running time of the shortest circuit basis algorithm presented is $O(\nu^3+\nu(n(m+n\log n))) = O(m^3+mn^2\log n)$. The following theorem summarizes this discussion.

Theorem 7.2 *Solving the shortest circuit basis problem (7.1) requires $O(\nu^3)$ time plus the time required to solve ν instances of the shortest odd circuit problem (7.2).* ∎

7.3. Bases of circuits and signed graphs

In this section we shall study a little bit further the relation between bases of circuits and odd circuits in signed graphs. Let $G = (V, E)$ be an undirected connected graph and let l be a nonnegative length function on E.

Assumption. We fix the edge set $\{e_1, \ldots, e_\nu\}$ of a spanning cotree of G and assume that all the signs we shall consider are a subset of $\{e_1, \ldots, e_\nu\}$ (as in the algorithm).

Lemma 7.4 *Let $\mathcal{B}' := \{C'_1, \ldots, C'_\nu\}$ be a shortest circuit basis of G. Then there exist signs $\Sigma'_1, \ldots, \Sigma'_\nu$ so that, after a possible relabeling of the circuits, the following holds for $k = 1, \ldots, \nu$:*

$$\begin{aligned} &(i)\ e_k \in \Sigma'_k \text{ and } e_k \notin \Sigma'_i\ (i = 1, \ldots, k-1); \\ &(ii)\ C'_k \text{ is a shortest odd circuit in } (G, \Sigma'_k); \\ &(iii)\ C'_k \text{ is an even circuit in } (G, \Sigma'_i)\ (i = k+1, \ldots, \nu). \end{aligned} \tag{7.4}$$

Note that by (i) of (7.4) the signs $\Sigma'_1, \ldots, \Sigma'_\nu$ are automatically independent. Also observe that the basis $\{C_1, \ldots, C_\nu\}$ returned by the shortest circuit basis algorithm and the signs $\Sigma_{1,1}, \ldots, \Sigma_{\nu,\nu}$ satisfy (7.4). Thus, the content of Lemma 7.4 is that any shortest circuit basis of G can be the algorithm's output.

Proof of Lemma 7.4. Consider the execution of the shortest circuit basis algorithm. Suppose that at beginning of the iteration k ($k < \nu$) of Step 2 of the algorithm we have that $C_1 = C'_1, C_2 = C'_2, \ldots, C_{k-1} = C'_{k-1}$. By Lemma 7.3, C'_i is an even circuit in $(G, \Sigma_{k,j})$ ($i = 1, \ldots, k-1$; $j = k, \ldots, \nu$). Let C_k be a shortest odd circuit in $(G, \Sigma_{k,k})$. Since $\mathcal{B}'$ is a basis of circuits we have that

$$C_k \;=\; C'_{i_1} \,\Delta\, C'_{i_2} \,\Delta\, \cdots \,\Delta\, C'_{i_{t_k}}$$

for distinct circuits $C'_{i_1}, \ldots, C'_{i_{t_k}}$ of $\mathcal{B}'$. There exists an index $i_{k'} \geq k$ so that $C'_{i_{k'}}$ is an odd circuit in $(G, \Sigma_{k,k})$. Actually, $C'_{i_{k'}}$ is a shortest odd circuit in $(G, \Sigma_{k,k})$

(otherwise $(\mathcal{B} \setminus \{C'_{i_{k'}}\}) \cup \{C_k\}$ would be a circuit basis shorter than $\mathcal{B}'$). Thus, by simply relabeling the circuits we may assume that $i_{k'} = k$ and $C_k = C'_k$. Applying the above reasoning repeatedly we eventually obtain $\{C'_1, \ldots, C'_\nu\}$ as output of the algorithm. $\Sigma'_1 := \Sigma_{1,1}, \Sigma'_2 := \Sigma_{2,2}, \ldots, \Sigma'_\nu := \Sigma_{\nu,\nu}$ are the desired signs. ∎

A statement analogous to the one in Lemma 7.4 holds if instead of a circuit basis we consider independent circuits of G. More specifically:

Lemma 7.5 *Let $C_1, \ldots, C_t$ be independent circuits of G. Then there exist independent signs $\Sigma_1, \ldots, \Sigma_t$ so that the following holds for $k = 1, \ldots, t$:*

(i) C_k is a odd circuit in (G, Σ_k); and
(ii) C_k is an even circuit in (G, Σ_i) $(i = k+1, \ldots, t)$.

Moreover, if $t < \nu$ then there exists a sign Σ so that C_k is an even circuit in (G, Σ). ∎

Lemma 7.6 *Let $C_1, \ldots, C_k$ be independent circuits and let $\Sigma_1, \ldots, \Sigma_t$ be independent signs of G so that C_i is an even circuit in (G, Σ_j) $(i = 1, \ldots, k;\ j = 1, \ldots, t)$. Then $k + t \leq \nu$.*

Proof. Let C_{k+1} be an odd circuit in (G, Σ_1) and, for $i = 2, \ldots, t$, let

$$\Sigma'_i := \begin{cases} \Sigma_i \,\Delta\, \Sigma_1 & \text{if } C_{k+1} \text{ is an odd} \\ & \text{circuit in } (G, \Sigma_i); \\ \Sigma_i & \text{otherwise.} \end{cases}$$

By Lemma 7.2 it follows that $C_1, \ldots, C_{k+1}$ are independent circuits and, by Lemma 7.1, we have that C_i is an even circuit in (G, Σ'_j) $(i = 1, \ldots, k+1;\ j = 2, \ldots, t)$. Applying this process repeatedly we eventually get independent circuits $C_1, \ldots, C_k, C_{k+1}, \ldots, C_{k+t}$. ∎

We shall use twice the following lemma.

Lemma 7.7 *Let $\{C'_1, \ldots, C'_\nu\}$ be a shortest circuit basis of G, let $\Sigma_1, \ldots, \Sigma_\nu$ be independent signs and let $C_1, \ldots, C_\nu$ be circuits of G so that C_i is a odd circuit in (G, Σ_i). Suppose that*

$$l(C'_1) \leq l(C'_2) \leq \ldots \leq l(C'_\nu) \text{ and } l(C_1) \leq l(C_2) \leq \ldots \leq l(C_\nu). \tag{7.5}$$

Then $l(C_k) \leq l(C'_k)$ $(k = 1, \ldots, \nu)$.

Proof. First we prove the following claim.

Claim 7.1 *For each $k = 1, \ldots, \nu$ there exist indices i_k and j_k, $1 \leq i_k \leq k \leq j_k \leq \nu$ so that C'_{i_k} is an odd circuit in (G, Σ_{j_k}).*

Proof. Suppose by contradiction that for some $k \in \{1, \ldots, \nu\}$ we have that C'_i is an even circuit in (G, Σ_j) $(i = 1, \ldots, k;\ j = k, \ldots, \nu)$. Thus, $C'_1, \ldots, C'_k$ and $\Sigma_k, \ldots, \Sigma_\nu$ are independent circuits and signs. Hence, by Lemma 7.6, $k + (\nu - k + 1) > \nu$, which is absurd. ∎

Now, let $k \in \{1, \ldots, \nu\}$. Choose indices i_k and j_k satisfying the conditions of Claim 7.1. Since C_{j_k} is a shortest odd circuit in (G, Σ_{j_k}) it follows that $l(C_{j_k}) \leq l(C'_{i_k})$. Combining the latter inequality with (7.5) we obtain

$$l(C_k) \leq l(C_{j_k}) \leq l(C'_{i_k}) \leq l(C'_k).$$

This completes the proof of Lemma 7.7. ∎

Theorem 7.3 *The length of a shortest circuit basis of G is equal to*

$$\max_{\Sigma_1, \ldots, \Sigma_\nu} \sum_{i=1}^{\nu} \{l(C_i) \mid C_i \text{ is a shortest odd circuit in } (G, \Sigma_i)\}, \tag{7.6}$$

where the maximum ranges over all independent signs $\Sigma_1, \ldots, \Sigma_\nu$ of G.

Proof. It follows from Lemma 7.4 that the maximum in (7.6) is greater than or equal to the length of a shortest circuit basis of G. Thus, it remains to prove that the maximum is less than or equal to the length of a shortest circuit basis.

Let $\{C'_1, \ldots, C'_\nu\}$ be a shortest circuit basis of G, let $\Sigma_1, \ldots, \Sigma_\nu$ be independent signs and let $C_1, \ldots, C_\nu$ be circuits so that C_i is a shortest odd circuit in (G, Σ_i). (Note that $\{C_1, \ldots, C_\nu\}$ is not necessarily a circuit basis.) We may assume that

$$l(C'_1) \leq l(C'_2) \leq \ldots \leq l(C'_\nu) \text{ and } l(C_1) \leq l(C_2) \leq \ldots \leq l(C_\nu).$$

By Lemma 7.7 it follows that $l(C_k) \leq l(C'_k)$ $(k = 1, \ldots, \nu)$. This gives to us the desired inequality. ∎

Theorem 7.4 *Let $\mathcal{B}$ be a set of circuits of G. Then the following are equivalent:*

(i) $\mathcal{B}$ is a shortest circuit basis of G;
(ii) $\mathcal{B}$ is a minimal set of circuits of G so that $\mathcal{B}$ contains a shortest odd circuit in (G, Σ) for every sign Σ of G.

Remark 7.1. An analogous statement also holds if we replace shortest circuit basis and shortest odd circuit by circuit basis and odd circuit, respectively.

Proof of Theorem 7.4. $(i) \Rightarrow (ii)$. It is clear that each shortest circuit basis contains a shortest odd circuit in (G, Σ) for every sign Σ of G. The fact that shortest circuit basis is a minimal set with this property follows immediately from Lemma 7.5.

$(ii) \Rightarrow (i)$. Let $\mathcal{B} := \{C_1, \ldots, C_t\}$, satisfying (ii). By the minimally of $\mathcal{B}$ it follows that $C_1, \ldots, C_t$ are independent circuits and from Lemma 7.5 we have that $\mathcal{B}$ must be a circuit basis. Also by Lemma 7.5 there are independent signs $\Sigma_1, \ldots, \Sigma_\nu$ so that C_i is an odd circuit in (G, Σ_i) $(i = 1, \ldots, \nu)$. We may assume that C_i is a shortest odd circuit of (G, Σ_i) $(i = 1, \ldots, \nu)$ (this can be proved analogously to the proof of Lemma 7.4). Hence, by Theorem 7.3, $\mathcal{B}$ is a shortest circuit basis. (We could equivalently have proved that any set of circuits satisfying (ii) can be the algorithm's output.) ∎

Lemma 7.8 *Let Σ and Σ' be signs of G and let C and C' be circuits of G so that:*

(i) C is a shortest odd circuit in (G, Σ) and an even circuit in (G, Σ');
(ii) C' is a shortest odd circuit in (G, Σ') and an odd circuit in (G, Σ).

Then C is a shortest odd circuit in $(G, \Sigma \,\Delta\, \Sigma')$.

Proof. C is clearly an odd circuit in $(G, \Sigma \,\Delta\, \Sigma')$. Let D be a shortest odd circuit in $(G, \Sigma \,\Delta\, \Sigma')$. Thus, by Lemma 7.1, D is either an odd circuit in (G, Σ) or in (G, Σ'). If D is an odd circuit in (G, Σ) we are done. Now, if D is an odd circuit in (G, Σ') then $l(D) \geq l(C') \geq l(C)$. ∎

Lemma 7.9 *Let $\mathcal{B}' := \{C'_1, \ldots, C'_\nu\}$ be a shortest circuit basis of G. Then there exist independent signs $\Sigma'_1, \ldots, \Sigma'_\nu$ so that the following holds for $k = 1, \ldots, \nu$:*

(i) C'_k is a shortest odd circuit in (G, Σ'_k); and
(ii) C'_k is an even circuit in (G, Σ'_i) $(i = 1, \ldots, \nu;\ i \neq k)$.

Remark 7.2. An analogous statement also holds if we replace shortest circuit basis and shortest odd circuit by circuit basis and odd circuit, respectively.

Proof of Lemma 7.9. Consider the shortest circuit basis algorithm presented with the following small modification in the define-part of Step 2:

Define for $i = 1, \ldots, \nu$:

$$\Sigma_{k+1,i} := \begin{cases} \Sigma_{k,i} \,\Delta\, \Sigma_{k,k} & \text{if } C_k \text{ is an odd} \\ & \text{circuit in } (G, \Sigma_{k,i}); \\ \Sigma_{k,i} & \text{otherwise.} \end{cases}$$

Thus, we are now defining $\Sigma_{k,i}$ $(k = 1, \ldots, \nu;\ i = 1, \ldots, \nu)$. We recall that any shortest circuit basis can be the output of the algorithm and consider a execution of the algorithm that returns $\mathcal{B}'$. We may assume that C'_k is the shortest circuit found during iteration k of Step 2 $(k = 1, \ldots, \nu)$. We claim that the independent signs $\Sigma'_1 := \Sigma_{\nu,1}, \Sigma'_2 := \Sigma_{\nu,2}, \ldots, \Sigma'_\nu := \Sigma_{\nu,\nu}$ fulfill the

lemma's conditions. Indeed, analogously to Lemma 7.3, one can prove that C'_k is an even circuit in $(G, \Sigma_{t,i})$ $(1 \leq k \leq t \leq \nu;\ i = 1, \ldots, \nu;\ i \neq k)$. Moreover, from the definition of $\Sigma_{k,i}$ and by Lemma 7.8 it follows that C'_k is a shortest odd circuit in $(G, \Sigma_{t,k})$ $(1 \leq k \leq t \leq \nu)$. ∎

Theorem 7.5 *Let $\mathcal{B} := \{C_1, \ldots, C_\nu\}$ be a set of circuits of G. Then the following are equivalent:*

(i) $\mathcal{B}$ is a shortest circuit basis of G;
(ii) there exist independent signs $\Sigma_1, \ldots, \Sigma_\nu$ so that the following holds for $k = 1, \ldots, \nu$:
(a) C_k is a shortest odd circuit in (G, Σ_k); and
(b) C_k is an even circuit in (G, Σ_i) $(i = 1, \ldots, \nu;\ i \neq k)$.

Remark 7.3. An analogous statement also holds if we replace shortest circuit basis and shortest odd circuit by circuit basis and odd circuit, respectively.

Proof of Theorem 7.5. $(i) \Rightarrow (ii)$. This is the contents of Lemma 7.9. $(ii) \Rightarrow (i)$. By Theorem 7.3 and $(i)(a)$ it suffices to prove that $C_1, \ldots, C_\nu$ are independent sets. Suppose by contradiction that (say) $C_\nu \in \langle\{C_1, \ldots, C_{\nu-1}\}\rangle$. This implies that $\{C_1, \ldots, C_{\nu-1}\}$ contains an odd circuit in (G, Σ_ν) contradicting $(ii)(b)$. ∎

From Theorem 7.5 and the proof of Lemma 7.7 we derive the following corollary.

Corollary 7.1 *Let $\{C_1, \ldots, C_\nu\}$ be a circuit basis of G and let $\{C'_1, \ldots, C'_\nu\}$ be a shortest circuit basis of G. Suppose that*

$$l(C'_1) \leq \ldots \leq l(C'_\nu) \text{ and } l(C_1) \leq \ldots \leq l(C_\nu).$$

Then $l(C'_k) \leq l(C_k)$ $(i = 1, \ldots, \nu)$.

Proof. By Theorem 7.5 there exist independent signs $\Sigma_1, \ldots, \Sigma_\nu$ so that C_k is an odd circuit in (G, Σ_k) $(k = 1, \ldots, \nu)$. Hence, by Lemma 7.7 we have that $l(C'_k) \leq l(C_k)$ $(i = 1, \ldots, \nu)$. ∎

Remark 7.4. It follows from Corollary 7.1 that the product in (6.8) is minimized if $\mathcal{B}'$ is a shortest circuit basis of the graph G_D.

7.4. Properties of circuits in a shortest circuit basis—Horton's algorithm

The first polynomial-time algorithm for the shortest circuit basis problem has been derived by Horton [1987]. His major observation was the following theorem.

Theorem 7.6 (Horton [1987]) *Let C be a circuit in a shortest circuit basis of G, let u and v be two vertices in C. Let $P_{u,v}$ and $P_{v,u}$ be the two $u - v$-paths in C. Then $P_{u,v}$ or $P_{v,u}$ is a shortest $u - v$-path in G.*

Proof. By Lemma 7.4 , C is a shortest odd circuit in (G, Σ) for some sign Σ. Suppose that neither $P_{u,v}$ nor $P_{v,u}$ is a shortest $u - v$-path and let P be a shortest $u - v$-path. Then, either $P \Delta P_{u,v}$ or $P \Delta P_{v,u}$ contains an odd circuit in (G, Σ) shorter than C, a contradiction. ∎

From Theorem 7.6 one can easily derive the following.

Theorem 7.7 (Horton [1987]) *Let C be a circuit of a shortest circuit basis of G and let r be a vertex of C. Then there exists an edge $\{u, v\}$ of C so that C consists of a shortest $r - u$-path, a shortest $v - r$-path and the edge $\{u,v\}$.* ∎

Horton proved that the following collection of cycles contains a shortest circuit basis of G[1]:

$$P_{r,u} \cdot e \cdot P_{v,r} \qquad (r \in V;\ e = \{u, v\}), \tag{7.7}$$

where $P_{x,y}$ denotes a shortest $x - y$-path in G. Horton's algorithm consists of constructing the polynomial length list of circuits in the collection (7.7) and then applying the greedy algorithm to this list in order to find a shortest circuit basis. As noted by Horton this greedy step is really necessary as the converse of Theorem 7.7 does not hold (see Figure 7.1).

Horton's algorithm

Step 1. For each vertex r of G find a shortest-path tree T_r with root r. Let $\mathcal{C}_r := \{C(T_r, e) \mid e \in E(G) \setminus E(T_r),\ r \in C(T_r, e)\}$ and let

$$\mathcal{F} := \cup_{r \in V(G)} \mathcal{C}_r.$$

Step 2. Apply the greedy algorithm to $\mathcal{F}$.

One can show that the set $\mathcal{F}$ defined in the algorithm contains a shortest circuit basis (this is obvious if we use perturbation or lexicographic techniques in order to find the shortest-path trees; in this case each shortest $u - v$-path in G is unique). The number of circuits in $\mathcal{F}$ is proportional to νn. Each circuit of $\mathcal{F}$ can be processed by the greedy algorithm in time $O(m\nu)$ (the time for the Gaussian elimination). Thus, the overall running time of Horton's algorithm is $O(m\nu^2 n) = O(m^3 n)$.

7.5. Bases of circuits and polyhedral theory

Let $G = (V, E)$ be an undirected 2-connected graph. For each subset $B \subseteq E$, we define its *incidence vector* $\chi^B = (\chi^B_e)_{e \in E}$ by

$$\chi^B_e := \begin{cases} 1 & \text{if } e \in B; \\ 0 & \text{otherwise.} \end{cases}$$

[1] The similarity between (2.4), (7.7) and (7.30) is noteworthy.

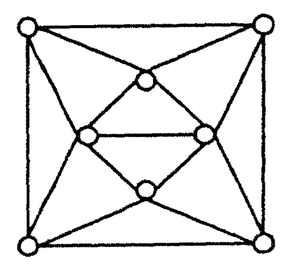


Figure 7.1: The circuit on the boundary of the graph satisfies the conditions of Theorem 7.7, but it is not contained in *the* shortest circuit basis.

For each circuit basis $\mathcal{B} := \{C_1, \ldots, C_\nu\}$ of G we shall denote by $x^{\mathcal{B}}$ the vector

$$\chi^{C_1} + \cdots + \chi^{C_\nu}.$$

Since for nonnegative length functions we can solve the shortest circuit basis problem in polynomial time it follows that we can minimize nonnegative length functions over the dominant of the convex hull of

$$\{x^{\mathcal{B}} \mid \mathcal{B} \text{ is a circuit basis of } G\}$$

in polynomial time as well (cf. Grötschel, Lovász & Schrijver [1981]). Hence, by the ellipsoid method, for every inequality $c^T x \geq b$ valid for this dominant we can find (in polynomial time) inequalities $a_1^T x \geq b_1, \ldots, a_k^T x \geq b_k$ defining facets of this dominant so that $c = a_1 + \cdots + a_k$ and $b \leq b_1 + \cdots + b_k$. So, it is not unreasonable to ask ourselves about an explicit description of this dominant by a system of linear inequalities. In what follows we make a few remarks concerning this question.

Let $P(G)$ denote the above polyhedron, i.e.,

$$P(G) \quad := \quad \text{conv.hull}\{x^{\mathcal{B}} \mid \mathcal{B} \text{ is a circuit basis of } G\} + \mathbb{R}_+^E. \tag{7.8}$$

To begin with, it is clear that the *trivial inequalities*

$$x_e \geq 1, \quad \text{for all } e \in E \tag{7.9}$$

define facets of $P(G)$.

If $\{C_1, \ldots, C_\nu\}$ is a circuit basis of G then $\{C_1 - \{e\}, \ldots, C_\nu - \{e\}\}$ is a basis of the circuit space of G/e. From this one derives that the *cut inequalities*

$$x(\delta_G(S)) \geq 2(d_G(S) - 1), \quad \text{for all cuts } \delta_G(S) \subset E, \tag{7.10}$$

are valid for $P(G)$. (Recall that if $\delta_G(S)$ is a cut then both $G[S]$ and $G[V \setminus S]$ are connected.) Observe that if $\delta_G(S) = \{e, f\}$ is a cut then the corresponding cut inequality is implied by the trivial inequalities $x_e \geq 1$ and $x_f \geq 1$. Let us define

$$Q(G) \quad := \quad \{x \in \mathbb{R}^E \mid x \text{ satisfies (7.9) and (7.10)}\}. \tag{7.11}$$

Clearly, $Q(G) \supseteq P(G)$. When does equality hold? This seems to be a question that should be addressed before the question concerning an explicit description of $P(G)$ by a system of linear inequalities.

Lemma 7.10 *Let $G = (V, E)$ be a graph and let $l \in \mathbb{Q}^E$ so that $l^T x \geq b$ defines a nontrivial facet of $P(G)$. Then $l_e \geq 0$ for all edges $e \in E$ and for each circuit C and $e \in C$, $l_e \leq l(C - \{e\})$.*

Proof. Suppose first that $l_f < 0$ for some $f \in E$. If $x \in P(G)$ is so that $l^T x = b$ then $x' := x + \chi^{\{f\}} \in P(G)$ and $l^T x' < b$. Thus, $l_e \geq 0$ for all $e \in E$.

Suppose now that $l_f > l(C - \{f\})$ for some circuit C of G with $f \in C$. We claim that $l^T x \geq b$ is a positive multiple of $x_f \geq 1$. To prove this we show that if $\mathcal{B}$ is a circuit basis of G satisfying $l^T x^{\mathcal{B}} = b$ then $x_f^{\mathcal{B}} = 1$, i.e. there is a unique circuit of $\mathcal{B}$ containing f. For, suppose there are at least two circuits of $\mathcal{B}$ containing f. We show that this implies the existence of a basis $\mathcal{B}'$ of the circuit space of G so that

$$l^T x^{\mathcal{B}'} \quad < \quad l^T x^{\mathcal{B}} = b \tag{7.12}$$

contradicting that $l^T x \geq b$ is a valid inequality of $P(G)$.

Since $\mathcal{B}$ is a circuit basis we have that

$$C \quad = \quad C_1 \,\Delta\, \cdots \,\Delta\, C_t, \quad C_i \in \mathcal{B}\ (i = 1, \ldots, t).$$

If $l(C) < l(C_j)$ for some $j \in \{1, \ldots, t\}$ then $\mathcal{B}' := (\mathcal{B} \setminus \{C_j\}) \cup \{C\}$ is a circuit basis satisfying (7.12). Thus, we assume that

$$l(C) \quad \geq \quad l(C_i) \quad (i = 1, \ldots, t). \tag{7.13}$$

We may also assume that there exists no circuit $C' \in \mathcal{B} \setminus \{C_1, \ldots, C_t\}$ containing f; otherwise $\mathcal{B}' := (\mathcal{B} \setminus \{C'\}) \cup \{C \,\Delta\, C'\}$ is a circuit basis fulfilling (7.12). Hence, $\{C_1, \ldots, C_t\}$ contains at least two circuits (say) C_1 and C_2 containing f. From (7.13) it follows that $l_f > l(C_i - \{f\})$ $(i = 1, 2)$. Therefore, $\mathcal{B}' := (\mathcal{B} \setminus \{C_1\}) \cup \{C_1 \,\Delta\, C_2\}$ is a basis of the circuit space of G satisfying (7.12). ∎

Theorem 7.8 *Let $G = (V, E)$ be a 2-connected planar graph with collection of cuts $\mathcal{C}^*$ and let $l \in \mathbb{Q}^E$ so that $l^T x \geq b$ defines a nontrivial facet of $P(G)$. Then l is a vector in the cut cone of G, i.e. there exists a function $\lambda : \mathcal{C}^* \to \mathbb{Q}_+$ so that*

$$l = \sum_{C^* \in \mathcal{C}^*} \lambda(C^*) \chi^{C^*}.$$

Theorem 7.8 is an easy consequence of Lemma 7.10 and the following theorem.

Theorem 7.9 (Seymour [1979]) *Let $G = (V, E)$ be a graph with collection of circuits $\mathcal{C}$. Let $p \in \mathbb{Q}_+^E$ so that for each cut B and $e \in B$, $p_e \leq p(B - \{e\})$. Then there exists $\lambda : \mathcal{C} \to \mathbb{Q}_+$ so that*

$$\sum_{C \in \mathcal{C}} \lambda(C)\chi^C = p.$$

∎

For a short proof of Theorem 7.9 see Coullard & Pulleyblank [1989].

Proof of Theorem 7.8. By applying Lemma 7.10 and Theorem 7.9 to the geometric dual $G^* = (\mathcal{F}, E^*)$ of G and the vector $l^* \in \mathbb{Q}^{E^*}$ associated with l we obtain that l is a vector in the cut cone of G. (If G is planar, the cuts of G correspond to the circuits of its geometric dual G^*.) ∎

Corollary 7.2 *Let $G = (V, E)$ be a planar graph. Then the cut inequalities (7.10) define facets of $P(G)$ for all cuts $\delta_G(S)$ with $d_G(S) \geq 3$.*

Proof. By Theorem 7.8 it suffices to show that for each cut $\delta_G(S)$ there exists a circuit basis $\mathcal{B}$ of G so that $x^{\mathcal{B}}(\delta_G(S)) = 2(d_G(S) - 1)$.

Let $\delta_G(S)$ be a cut of G and let T be a spanning tree of G so that $|E(T) \cap \delta_G(S)| = 1$. The fundamental circuits of G with respect to T (i.e. $\{C(T, e) \mid e \in E(G) \setminus \delta_G(S)\}$) form a circuit basis $\mathcal{B}$ so that $x^{\mathcal{B}}(\delta_G(S)) = 2(d_G(S) - 1)$. ∎

Despite Theorem 7.8 and Corollary 7.2, it is not necessarily true that if G is a planar graph then $Q(G) = P(G)$. For example, let $W = (V, E)$ be the graph shown in Figure 7.2. $Q(W)$ is an integral polyhedron (cf. Theorem 7.11) and the vector v formed by the numbers next to the edges in the figure is a vertex of $Q(W)$. However, this vertex of $Q(W)$ does not correspond to any circuit basis of W. (Note that $\nu(W) = 5$ and that $\sum_{e \in E} v_e = 15$. So, if there exists a circuit basis of W corresponding to v this basis must be formed by triangles. One can verify that such a basis does not exist.)

In the sequel we observe that $Q(G) = P(G)$ does hold if G is a wheel. To prove this we show first that $Q(G)$ is integral if G has a "Halin graph" as geometric dual. (We will define Halin graphs below.) Next, using this integrality we show that $Q(G) = P(G)$ if G is a wheel.

Integrality of $Q(G)$ for dual graphs of Halin graphs

Let $G = (V, E)$ be graph with collection of cuts $\mathcal{C}^*$. It follows from a result due to Edmonds & Giles [1977] (see also Schrijver [1986: Thm. 22.1 and Cor. 22.1a])) that $Q(G)$ is an integral polyhedron if and only if

$$\min\{l^T x \mid x \in Q(G)\} \tag{7.14}$$

is an integer for each nonnegative integral vector l.

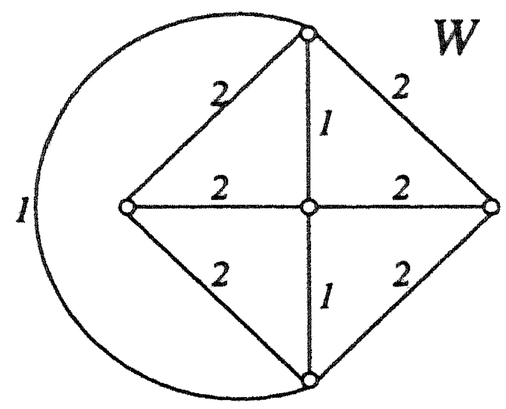


Figure 7.2: A planar graph W so that $Q(W) \neq P(W)$.

Lemma 7.11 *We may assume that (7.14) has no optimal solution x^* so that $x^*_f = 1$ for some $f \in E$.*

Proof. Let $f \in E$ so that $x^*_f = 1$. Thus, the optimum in (7.14) is an integer if an only if

$$\min\{(l^f)^T x \mid x \in Q(G - \{f\})\}$$

is an integer, where l^f denotes the restriction of l to $E - \{f\}$. ∎

In particular, it follows from Lemma 7.11 that l in (7.14) maybe assumed to belong to the cut cone of G. Indeed, By Theorem 7.9 if l is not a vector in the cut cone of G then there exist a circuit C and an edge $f \in C$ so that $l_f > l(C - \{f\})$. From this one can derive that any optimal solution x^* of (7.14) satisfies $x^*_f = 1$ (otherwise $x' \in \mathbb{Q}^E$ defined as:

$$x'_e := \begin{cases} 1 & \text{if } e = f; \\ x^*_e + x^*_f - 1 & \text{if } e \in C - \{f\}; \\ x^*_e & \text{otherwise,} \end{cases}$$

is a vector in $Q(G)$ so that $l^T x' < l^T x^*$, a contradiction).

Consider now the following linear programming problem dual of problem (7.14):

$$\begin{aligned} \max\{\sum_{e \in E} z_e + 2 \cdot \sum_{\delta_G(S) \text{ cut}} y_S(d_G(S) - 1) \mid \\ \sum_{e \in E} z_e \chi^{\{e\}} + \sum_{\delta_G(S) \text{ cut}} y_S \chi^{\delta_G(S)} = l, \qquad (7.15) \\ z_e \geq 0 \text{ for all edges } e \in E; \\ y_S \geq 0 \text{ for all cuts } \delta_G(S) \subset E\}. \end{aligned}$$

By the duality theorem of linear programming the minimum in (7.14) equals the maximum in (7.15). We show that the maximum is an integer.

Lemma 7.12 *Let (z^*, y^*) be an optimum solution in (7.15). Then we may assume that $z_e^* = 0$ for all edges $e \in E$.*

Proof. The lemma follows from Lemma 7.11 by complementary slackness. ∎

Lemma 7.13 *The maximum in (7.15) is an integer if and only if*

$$\min\{ \sum_{\delta_G(S) \text{ cut}} y_S \mid \sum_{\delta_G(S) \text{ cut}} y_S \chi^{\delta_G(S)} = l, \quad y_S \geq 0 \text{ for all cuts } \delta_G(S) \subset E\}$$

is a half-integer.

Proof. By Lemma 7.12 we have that the maximum in (7.15) is equal to

$$\begin{aligned} &2 \cdot \max\{ \sum_{\delta_G(S) \text{ cut}} y_S(d_G(S) - 1) \mid \sum_{\delta_G(S) \text{ cut}} y_S \chi^{\delta(S)} = l, \quad y_S \geq 0 \text{ for all cuts } \delta_G(S) \subset E\} \\ &= 2 \cdot l(E) - 2 \cdot \min\{ \sum_{\delta_G(S) \text{ cut}} y_S \mid \sum_{\delta_G(S) \text{ cut}} y_S \chi^{\delta(S)} = l, \quad y_S \geq 0 \text{ for all cuts } \delta_G(S) \subset E\}. \end{aligned}$$

The lemma follows immediately from the above relation. ∎

A *Halin graph* $G = (V, T \cup C)$ is a planar graph consisting of a tree T that has no degree two vertices, together with a simple circuit C whose vertices are the degree-one vertices of T (see Figure 7.3).

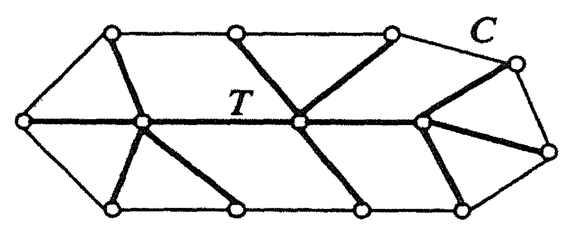


Figure 7.3: An example of a Halin graph $G = (V, T \cup C)$. The thicker lines represent the edges of the tree T and the thiner lines the edges of the circuit C.

Theorem 7.10 (Coullard & Pulleyblank [1989]) *Let $G = (V, T \cup C)$ be a Halin graph with collection of circuits $\mathcal{C}$ and $l \in \mathbb{Z}_+^{T \cup C}$. Then*

$$\min\{\sum_{C \in \mathcal{C}} y_C \mid \sum_{C \in \mathcal{C}} y_C \chi^C = l, \quad y_C \geq 0 \text{ for all circuits } C \in \mathcal{C}\}$$

is a half-integer. ∎

Remark 7.5. Actually, Coullard & Pulleyblank have given an explicit formula for the minimum in Theorem 7.10.

Combining Lemma 7.13 with Theorem 7.10 we obtain the following theorem.

Theorem 7.11 *Let $G = (V, E)$ be a graph having a Halin graph as geometric dual. Then $Q(G)$ is an integral polyhedron.* ∎

The relation $Q(G) = P(G)$ does hold for wheels

A *wheel* is a Halin graph $G = (V, T \cup C)$ for which the tree T is a star. Given a integer $n \geq 2$, W_n denotes the wheel consisting of a circuit with n vertices and one additional vertex, called the *center* (of the star), which is adjacent to each vertex of the circuit. The edges incident to the center are called *spokes* and the remaining edges and vertices are called *rim edges* and *rim vertices*, respectively.

Theorem 7.12 *Let $n \geq 2$ be an integer. Then $Q(W_n) = P(W_n)$.*

Proof. Let x^* be a vertex of $Q(W_n)$. By Theorem 7.11 we have that x^* is integral. We proceed by induction on n and show that there exists a circuit basis $\{C_1, \ldots, C_n\}$ of W_n so that

$$x^* = \chi^{C_1} + \cdots + \chi^{C_n}. \tag{7.16}$$

(Observe that $\nu(W_n) = n$.)

Let e_1, e_2 and f_1, f_2 be the spoke and rim edges of W_2, respectively. To see that $Q(W_2) = P(W_2)$ we first note that $x^*(\{e_1, f_1, f_2\}) = x^*(\{e_2, f_1, f_2\})$. Indeed, if (say) $x^*_{e_1} > x^*_{e_2}$ then the vector $x' \in \mathbb{Z}_+^{\{e_1, e_2, f_1, f_2\}}$ defined as

$$x'_e := \begin{cases} x^*_{e_2} & \text{if } e = e_1; \\ x^*_e & \text{otherwise,} \end{cases}$$

would be a vector in $Q(W_2)$ contradicting the hypothesis that x^* is a vertex of $Q(W_2)$ (recall that $Q(G)$ is a polyhedron of the blocking type). Similarly, from the integrality of the vertex x^* and from the fact that $x^*(\{e_1, f_1, f_2\}) = x^*(\{e_2, f_1, f_2\}) \geq 4$ we can easily derive that either $x^*_{e_1} = x^*_{e_2} = 2$ or $x^*_{f_1} = 2$ or $x^*_{f_2} = 2$ and the remaining components of x^* are equal to 1. From this one can promptly observe that there exists a (fundamental) circuit basis $\{C_1, C_2\}$ of W_2 fulfilling (7.16).

Now, let $W_n = (V, T \cup C)$ be a wheel with $n \geq 3$. Let

$$(v_1, f_1, v_2, f_2, \ldots, f_{n-1}, v_n, f_n, v_1)$$

be the rim circuit of W_n, where $v_1, v_2, \ldots, v_n$ are vertices of G, and $f_1, f_2, \ldots, f_n$ are rim edges of G and let $e_1, e_2, \ldots, e_n$ be the spoke edges of W_n, where $e_i = \{v, v_i\}$ $(i = 1, \ldots, n)$ (thus, v is the center of W_n, see Figure 7.4).

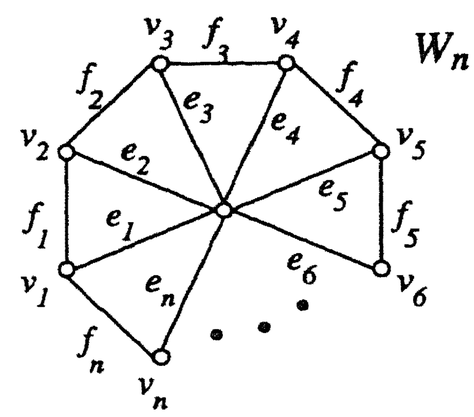


Figure 7.4: The wheel W_n.

Claim 7.2 *There exists a spoke edge e of W_n so that $x_e^* = 1$.*

Proof. Let $x' \in \mathbb{Z}_+^{T \cup C}$ defined as follows:

$$x'_e := \begin{cases} 2 & \text{if } e \in T; \\ 1 & \text{otherwise.} \end{cases}$$

We have that $x' \in P(W_n)$ (as $x' = \chi^{C_1} + \cdots + \chi^{C_n}$, where $\{C_1, \ldots, C_n\}$ is the circuit basis of W_n formed by the fundamental circuits of W_n with respect to the star T). Hence, if $x_e^* \geq 2$ for all spoke edges e of W_n then $x^* = x'$ and we are done (otherwise we would have that $x^* = x' + y$ with $0 \neq y \in \mathbb{Z}_+^{T \cup C}$ contradicting the hypothesis that x^* is a vertex of $Q(G)$). ∎

Since $x^* \in Q(G)$ and $n \geq 3$, $x^*(\delta_G(v)) \geq 2(n-1)$ implies the following claim.

Claim 7.3 *There exists a spoke edge e of W_n so that $x_e^* \geq 2$.* ∎

We shall say that a cut $\delta_G(S)$ is *tight* (with respect to x^*) if $x^*(\delta_G(S)) = 2(d_G(S) - 1)$.

Claim 7.4 *Let e be an edge of W_n with $x_e^* \geq 2$. Then there exists a tight cut $\delta_G(S_e)$ of W_n so that $e \in \delta_G(S_e)$.*

Proof. Suppose to the contrary, i.e. W_n has no tight cut containing e. Thus, $x^* - \chi^e \in Q(G)$ contradicting the fact that x^* is a vertex of $Q(G)$. ∎

Claim 7.5 *Let $e = \{u, v\}$ be a spoke edge so that $x_e^* = 1$ and let f and f' be distinct rim edges of W_n with u as an endpoint. Then $|x_f^* - x_{f'}^*| = 1$.*

Instead of proving Claim 7.5 we shall prove a claim a little bit stronger. Since $x^*(\delta_G(\{u\})) \geq 4$, we may assume that (say) $x_f^* \geq 2$. By Claim 7.4 there exists a tight cut $\delta_G(S_f)$ so that $f \in \delta_G(S_f)$. We prove the following claim that implies Claim 7.5.

Claim 7.6 *Let $e = \{u, v\}$ be a spoke edge so that $x_e^* = 1$ and let f and f' be the rim edges of W_n with u as an endpoint. Suppose that $x_f^* \geq 2$ and let $\delta_G(S_f)$ be a tight cut so that $f \in \delta_G(S_f)$. Then*

$$\begin{aligned}
&(i)\ f' \in \delta_G(S_f) \text{ and } e \in \delta_G(S_f) \Rightarrow x_f^* = 2 \text{ and } x_{f'}^* = 1;\\
&(ii)\ f' \in \delta_G(S_f) \text{ and } e \notin \delta_G(S_f) \Rightarrow x_f^* = 2 \text{ and } x_{f'}^* = 1;\\
&(iii)\ f' \notin \delta_G(S_f) \text{ and } e \notin \delta_G(S_f) \Rightarrow x_{f'}^* = x_f^* + 1; \text{ and}\\
&(iv)\ f' \notin \delta_G(S_f) \text{ and } e \in \delta_G(S_f) \Rightarrow |x_f^* - x_{f'}^*| = 1.
\end{aligned}$$

Proof. We consider each one of the 4 possibilities separately.

Case 1. $f' \in \delta_G(S_f)$ and $e \in \delta_G(S_f)$ (equivalently $\delta_G(S_f) = \delta_G(\{u\})$). In this case it is clear that $x_f^* = 2$ and $x_{f'}^* = 1$.

Case 2. $f' \in \delta_G(S_f)$ and $e \notin \delta_G(S_f)$ (equivalently $\delta_G(S_f) = \delta_G(\{u, v\})$).

We have that

$$\begin{aligned}
2(n-1) &\leq x^*(\delta_G(\{v\}))\\
&= x^*(\delta_G(S_f)) + x_e^* - x_{f'}^* - x_f^*\\
&= 2(d_G(S_f) - 1) + 1 - x_{f'}^* - x_f^*\\
&= 2(n-1) + 3 - x_{f'}^* - x_f^*.
\end{aligned}$$

Thus, $x_{f'}^* + x_f^* \leq 3$. On the other hand

$$x_f^* \geq 2, \quad \text{and} \quad x_{f'}^* \geq 1.$$

It follows that $x_f^* = 2$ and $x_{f'}^* = 1$.

Case 3. $f' \notin \delta_G(S_f)$ and $e \notin \delta_G(S_f)$

Let $S_f' \subset V$ so that $\delta_G(S_f') = (\delta_G(S_f) - \{f\}) \cup \{e, f'\}$. It follows that

$$\begin{aligned}
2(d_G(S_f') - 1) &\leq x^*(\delta_G(S_f'))\\
&= x^*(\delta_G(S_f)) - x_f^* + x_e^* + x_{f'}^*\\
&= 2(d_G(S_f) - 1) - x_f^* + 1 + x_{f'}^*\\
&= 2(d_G(S_f') - 1) - 2 - x_f^* + 1 + x_{f'}^*\\
&= 2(d_G(S_f') - 1) - 1 - x_f^* + x_{f'}^*.
\end{aligned}$$

Hence, $x_{f'}^* - 1 - x_f^* \geq 0$ implying that

$$x_{f'}^* \geq x_f^* + 1. \tag{7.17}$$

By (7.17) and Claim 7.4 we know that there exists a tight cut $\delta_G(S_{f'})$ so that $f' \in \delta_G(S_{f'})$. Assume that $u \in S_{f'}$. If $f \in \delta_G(S_{f'})$ then by symmetry

and Cases 1 and 2 above it follows that $x^*_{f'} = 2$ and $x^*_f = 1$, a contradiction. Thus $f \notin \delta_G(S_{f'})$.

Suppose that $e \notin \delta_G(S_{f'})$. Then by symmetry and similarly to the proof of (7.17) we can derive that $x^*_f \geq x^*_{f'} + 1$ contradicting (7.17). Therefore $e \in \delta_G(S_{f'})$. Let $S'_{f'} \subset V$ so that $\delta_G(S'_{f'}) = (\delta_G(S_{f'}) - \{e, f'\}) \cup \{f\}$. We have that

$$\begin{aligned} 2(d_G(S'_{f'}) - 1) &\leq x^*(\delta_G(S'_{f'})) \\ &= x^*(\delta_G(S_{f'})) + x^*_f - x^*_e - x^*_{f'} \\ &= 2(d_G(S_{f'}) - 1) + x^*_f - 1 - x^*_{f'} \\ &= 2(d_G(S'_{f'}) - 1) + 2 + x^*_f - 1 - x^*_{f'} \\ &= 2(d_G(S'_{f'}) - 1) + x^*_f + 1 - x^*_{f'}. \end{aligned}$$

Hence $x^*_{f'} \leq x^*_f + 1$. This latter inequality combined with (7.17) give us that $x^*_{f'} = x^*_f + 1$.

Case 4. $f' \notin \delta_G(S_f)$ and $e \in \delta_G(S_f)$.

Without loss of generality we assume that $f = f_n$, $f' = f_1$ and $e = e_1$ (thus $u = v_1$; see Figure 7.4). Similarly to what we have done in Cases 2 and 3 above one can derive that $x^*_{f_1} \geq x^*_{f_n} - 1$. Suppose that

$$x^*_{f_1} > x^*_{f_n} - 1. \tag{7.18}$$

We show that this assumption either leads to a contradiction or implies that $x^*_{f_1} = x^*_{f_n} + 1$. By Claim 7.4, there exists a tight cut $\delta_G(S_{f_1})$ of W_n containing f_1.

By symmetry we have the following implications:

(*i*) $f_n \in \delta_G(S_{f_1})$ and $e \in \delta_G(S_{f_1}) \Rightarrow x^*_{f_1} = 2$ and $x^*_{f_n} = 1$ (Case 1);
(*ii*) $f_n \in \delta_G(S_{f_1})$ and $e \notin \delta_G(S_{f_1}) \Rightarrow x^*_{f_1} = 2$ and $x^*_{f_n} = 1$ (Case 2);
(*iii*) $f_n \notin \delta_G(S_{f_1})$ and $e \notin \delta_G(S_{f_1}) \Rightarrow x^*_{f_n} = x^*_{f_1} + 1$ (Case 3).

Thus, we may assume that for all tight cuts $\delta_G(S_{f_1})$ containing f_1 the following holds:

$$f_n \notin \delta_G(S_{f_1}), \quad \text{and} \quad e_1 \in \delta_G(S_{f_1}). \tag{7.19}$$

Also by symmetry we may assume that for all tight cuts $\delta_G(S_{f_n})$ containing f_n the following holds:

$$f_1 \notin \delta_G(S_{f_n}) \quad \text{and} \quad e_1 \in \delta_G(S_{f_n}). \tag{7.20}$$

It follows from (7.19) (and again by symmetry) that $x^*_{f_n} \geq x^*_{f_1} - 1$, i.e.

$$x^*_{f_1} \leq x^*_{f_n} + 1. \tag{7.21}$$

We show that $x^*_{f_n}$ can not be equal to $x^*_{f_1}$; by (7.18) and (7.21) this completes the analysis of Case 4.

Let C be a maximum-length (with respect to the number of edges) circuit of W_n so that:

$$\begin{aligned} &(i)\ C \text{ traverses the rim edges } f_n, f_1 \text{ and the center } v \text{ of } W_n; \\ &(ii)\ x^*_e = 1 \text{ for all chord edges } e \text{ of } C. \end{aligned} \tag{7.22}$$

(An edge connecting two vertices of a circuit but not contained in the circuit is called a *chord*.) So, there exist indexes p_1 and p_2 so that

$$C = (v, e_{p_2}, v_{p_2}, f_{p_2}, \ldots, v_1, f_1, v_2, f_2, \ldots, f_{p_1-1}, v_{p_2}, e_{p_1}, v).$$

By (7.22) (ii) it follows that

$$x^*_{e_{p_2+1}} = \ldots = x^*_{e_1} = \ldots = x^*_{e_{p_1-1}} = 1.$$

(See Figure 7.5(a).)

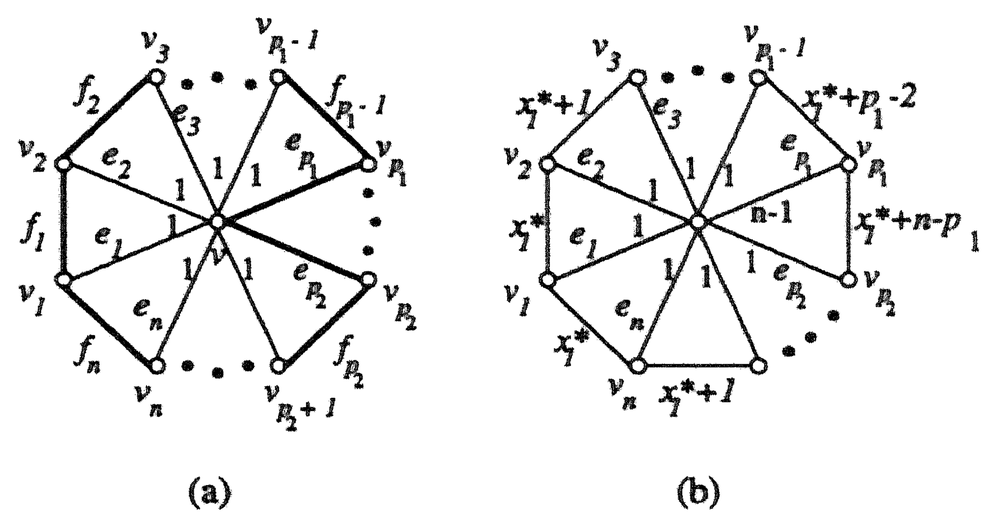


Figure 7.5: Figure (a) shows the circuit C defined in 7.22 in thicker lines. Figure (b) illustrates the situation when $x^*_{e_{p_1}} = 1$ or $x^*_{e_{p_2}} = 1$.

For a moment let us suppose that $x^*_{e_2} = 1$ (i.e. $p_1 \geq 3$). Let $\delta_G(S_{f_1})$ be a tight cut containing f_1. By Case 2 above if $f_2 \in \delta_G(S_{f_1})$ then $x^*_{f_1} = 2$ and $x^*_{f_2} = 1$. This implies that $\delta_G(\{v_2\})$ is a tight cut contradicting (7.19). Hence, by Case 3, $x^*_{f_2} = x^*_{f_1} + 1$ and for each tight cut $\delta_G(S_{f_2})$ containing f_2 we have that $\{e_1, e_2\} \subset \delta_G(S_{f_2})$. Invoking this reasoning repeatedly we obtain that for $i = 2, \ldots, p_1 - 1$ and all tight cuts $\delta_G(S_{f_i})$ containing f_i the following holds

$$x^*_{f_i} = x^*_{f_{i-1}} + 1 \ \text{ and } \ \{e_1, e_2, \ldots, e_i\} \subset \delta_G(S_{f_i}). \tag{7.23}$$

Similarly, by symmetry, we have that for $i = p_2 + 1, p_2 + 2, \ldots, n$ and all tight cuts $\delta_G(S_{f_i})$ containing f_i the following holds

$$x^*_{f_i} = x^*_{f_{i+1}} + 1 \ \text{ and } \ \{e_{i+1}, e_{i+2}, \ldots, e_1\} \subset \delta_G(S_{f_i}), \tag{7.24}$$

where $f_{n+1} := f_1$ and $e_{n+1} := e_1$. We claim now that $x^*_{e_{p_1}} \geq 2$ and $x^*_{e_{p_2}} \geq 2$. Indeed, if (say) $x^*_{e_{p_2}} = 1$ then by the maximality of the length of C it follows that $p_2 = p_1 + 1$. Thus, we have that $x^*_{p_1} \geq n-1$ (as $x^*(\{v\}) \geq 2(n-1)$). Since we are assuming $x^*_{f_n} = x^*_{f_1} \geq 2$, there exists no tight cut in W_n containing f_n or f_1 (see Figure 7.5(b)), a contradiction.

Let $x^1 \in \mathbb{Z}_+^{T \cup C}$ and $x^2 \in \mathbb{Z}_+^{T \cup C}$ defined as follows:

$$x^1_e := \begin{cases} x^*_e - 1 & \text{if } e \in \{e_{p_2}, f_{p_2}, f_{p_2+1}, \ldots, f_n\}; \\ x^*_e + 1 & \text{if } e \in \{f_1, f_2, \ldots, f_{p_1-1}, e_{p_1}\}; \\ x^*_e & \text{otherwise,} \end{cases}$$

$$x^2_e := \begin{cases} x^*_e - 1 & \text{if } e \in \{f_1, f_2, \ldots, f_{p_1-1}, e_{p_1}\}; \\ x^*_e + 1 & \text{if } e \in \{e_{p_2}, f_{p_2}, f_{p_2+1}, \ldots, f_n\}; \\ x^*_e & \text{otherwise.} \end{cases}$$

From (7.23), (7.24) and our assumption that $x^*_{f_n} = x^*_{f_1} \geq 2$ it follows that $x^1 \in Q(G)$ and $x^2 \in Q(G)$. However $x^* = (x^1 + x^2)/2$ contradicting the fact that x^* is a vertex of $Q(G)$. This completes the analysis of Case 4 and the proof of Claim 7.5. ∎

By Claims 7.2 and 7.3 we may assume without loss of generality that $x^*_{e_1} \geq 2$ and $x^*_{e_2} = 1$.

Claim 7.7 *We may assume that* $x^*_{f_1} = x^*_{f_2} + 1$.

Proof. By Claim 7.5 we have that $|x^*_{f_1} - x^*_{f_2}| = 1$. Let us suppose that $x^*_{f_2} = x^*_{f_1} + 1$ (otherwise we are done). Thus, $x^*_{f_2} \geq 2$ and by Claim 7.4 there exists a tight cut $\delta_G(S_{f_2})$ so that $f_2 \in \delta_G(S_{f_2})$. By Claim 7.6 we may assume that $e_2 \in \delta_G(S_f)$ (the unique impossible case of Claim 7.6 is (*iii*)—Claim 7.6 (*ii*) implies that $\delta_G(\{v_2\})$ is a tight cut).

Let f_k be the rim edge distinct of f_2 in the cut $\delta_G(S_{f_2})$, note that $k \geq 3$. We assume $x^*_{e_3} = 1$; otherwise, by symmetry, we are done. Let $p \geq 3$ be an integer so that $x^*_{e_2} = \ldots = x^*_{e_p} = 1$. We prove that $p < k$. For suppose that $x^*_{e_2} = \ldots = x^*_{e_k} = 1$. Then

$$\begin{aligned} 2(n-1) &\leq x^*(\delta_G(\{v\})) \\ &= x^*(\delta_G(S_{f_2})) - x^*_{f_2} - x^*_{f_k} + x^*(\{e_3, \ldots, e_k\}) \\ &= x^*(\delta_G(S_{f_2})) - x^*_{f_2} - x^*_{f_k} + (k-2) \\ &= 2(d_G(S_{f_2}) - 1) - x^*_{f_2} - x^*_{f_k} + (k-2) \\ &= 2(n-k+3) - x^*_{f_2} - x^*_{f_k} + (k-2) \\ &= 2n - k + 4 - x^*_{f_2} - x^*_{f_k}. \end{aligned}$$

Hence, $k + x^*_{f_2} + x^*_{f_k} \leq 6$ implying that $k = 3$, $x^*_{f_2} = 2$ and $x^*_{f_3} = 1$. Therefore

$$\begin{aligned} 6 &\leq x^*(\delta_G(\{v_2, v_3\})) \\ &= x^*_{f_1} + x^*_{e_2} + x^*_{e_3} + x^*_{f_3} \\ &= 4, \end{aligned}$$

a contradiction (recall that $x^*_{f_2} = x^*_{f_1} + 1$).

Let us assume that $v_2 \in S_{f_2}$. Since $e_2 \in \delta_G(S_{f_2})$ it follows that $e_3 \notin \delta_G(S_{f_2})$. If $f_3 \in \delta_G(S_{f_2})$ then by Claim 7.6 (ii) $x^*_{f_2} = 2$ and $x^*_{f_3} = 1$. However, this implies that $x^*(\delta_G(\{v_2, v_3\}) = 4 < 6 = 2(d_G(\{v_2, v_3\}) - 1)$. Hence, $f_3 \notin \delta_G(S_{f_2})$ and by Claim 7.6 (iii) it follows that $x^*_{f_3} = x^*_{f_2} + 1$ and therefore $\delta_G(S_{f_2} \cup \{v_3\})$ is a tight cut. Similarly, by successive invocations of Claim 7.6 (iii) we conclude that

$$x^*_{f_2} = x^*_{f_1} + 1,\ x^*_{f_3} = x^*_{f_2} + 1,\ \ldots\ , x^*_{f_p} = x^*_{f_{p-1}} + 1,$$

and

$$\delta_G(S_{f_2}),\ \delta_G(S_{f_2} \cup \{v_3\}),\ \ldots\ , \delta_G(S_{f_2} \cup \{v_3, \ldots, v_p\})$$

are all tight cuts.

Therefore, we have that $x^*_{e_p} = 1$, $x^*_{e_{p+1}} \geq 2$ and $x^*_{f_p} = x^*_{f_{p-1}} + 1$ which, by symmetry, completes the proof of our claim. ∎

We define $C_n := (v, e_1, v_1, f_1, v_2, e_2, v)$. Let $x'' := x^* - \chi^{C_n}$ and $G'' := W_n - \{e_2\}$. In view of Claim 7.7 we have that $x''_{f_1} = x''_{f_2}$. Let $G' = (V', E')$ be the graph obtained from G'' by replacing the path $(v_1, f_1, v_2, f_2, v_3)$ by an edge f' connecting v_1 and v_3 (thus $V' = V \setminus \{v_2\}$ and $E' = ((T \cup C) \setminus \{e_2, f_1, f_2\}) \cup \{f'\}$) and let $x' \in \mathbb{Z}^{E'}_+$ defined as follows:

$$x'_e := \begin{cases} x''_{f_1} & \text{if } e = f'; \\ x''_e & \text{otherwise.} \end{cases}$$

G' is isomorphic to W_{n-1} and $x' \in Q(G')$. To see the latter note that if $x' \notin Q(G')$ then there exists a cut $\delta_{G'}(S_{f'})$ so that $f' \in \delta_{G'}(S_{f'})$, $e_1 \in \delta_{G'}(S_{f'})$ and $x'(\delta_{G'}(S_{f'})) < 2(d_{G'}(S_{f'}) - 1)$. We assume that $v_1 \in S_{f'}$ and denote $S_{f'} \cup \{v_2\}$ by S_{f_2}. It follows that

$$\begin{aligned} x^*(\delta_G(S_{f'})) &= x'(\delta_{G'}(S_{f'})) + 2 \\ &< 2(d_{G'}(S_{f'}) - 1) + 2 \\ &= 2(d_G(S_{f'}) - 1) + 2, \end{aligned}$$

i.e. $x^*(\delta_G(S_{f'})) \leq 2(d_G(S_{f'}) - 1) + 1$. However, this implies that

$$\begin{aligned} 2(d_G(S_{f_2}) - 1) &\leq x^*(\delta_G(S_{f_2})) \\ &= x^*(\delta_G(S_{f'})) - x^*_{f_1} + x^*_{e_2} + x^*_{f_2} \\ &\leq 2(d_G(S_{f'}) - 1) + 1 - x^*_{f_1} + 1 + x^*_{f_2} \\ &= 2(d_G(S_{f_2}) - 1) - x^*_{f_1} + x^*_{f_2} \\ &= 2(d_G(S_{f_2}) - 1) - 1, \end{aligned}$$

contradicting the fact that $x^* \in Q(G)$. Therefore $x' \in Q(G')$. Actually, as x^* is a vertex of $Q(G)$ it follows that x' is a vertex of $Q(G')$. (Indeed, if $x' = \lambda y' + \mu z'$ with $y', z' \in Q(G')$ and $\lambda, \mu \in \mathbb{R}_+$, $\lambda + \mu = 1$ then $y^* := y' + \chi^{C_n}$ and $z^* := z' + \chi^{C_n}$ are vectors in $Q(G)$ so that $x^* = \lambda y^* + \mu z^*$, a contradiction.)

Since G' is isomorphic to W_{n-1} and x' is an integral vertex of $Q(G')$ it follows by the induction hypothesis that $Q(G') = P(G')$ and therefore there exists a circuit basis $\{C'_1, \ldots, C'_{n-1}\}$ of G' so that

$$x' = \chi^{C'_1} + \cdots + \chi^{C'_{n-1}}.$$

Replacing the occurrences of f' in each circuit $C'_1, \ldots, C'_{n-1}$ by the length-two path $(v_1, f_1, v_2, f_2, v_3)$ we obtain a set $\{C_1, \ldots, C_{n-1}\}$ of independent circuits of W_n. Now, it is clear that $\{C_1, \ldots, C_{n-1}, C_n\}$ is a circuit basis of W_n (C_n is the unique circuit containing the edge e_2) fulfilling (7.16), i.e.

$$x^* \;=\; \chi^{C_1} + \cdots + \chi^{C_n}.$$

With this we complete the proof of Theorem 7.12. ∎

Theorem 7.12 cannot be extended to all graphs G having a Halin graph as dual; despite $Q(G)$ be an integral polyhedron for those graphs. For example the graph W shown in Figure 7.2 has a Halin graph as dual and therefore $Q(W)$ is integral by Theorem 7.11. However, as we have already mentioned $Q(W) \neq P(W)$.

7.6. An application: cycle analysis of electrical networks

In *electric circuit theory* one is interested in modeling mathematically an electrical network. An *electrical network* is composed by the interconnection of two-terminal electric devises (i.e. each devise has two accessible terminals) as lamps, batteries and generators. We model lamps as *resistors*, batteries as *voltage sources* and generators as *current sources*. For an example of an electrical network see Figure 7.6. The interconnection structure of an electrical network

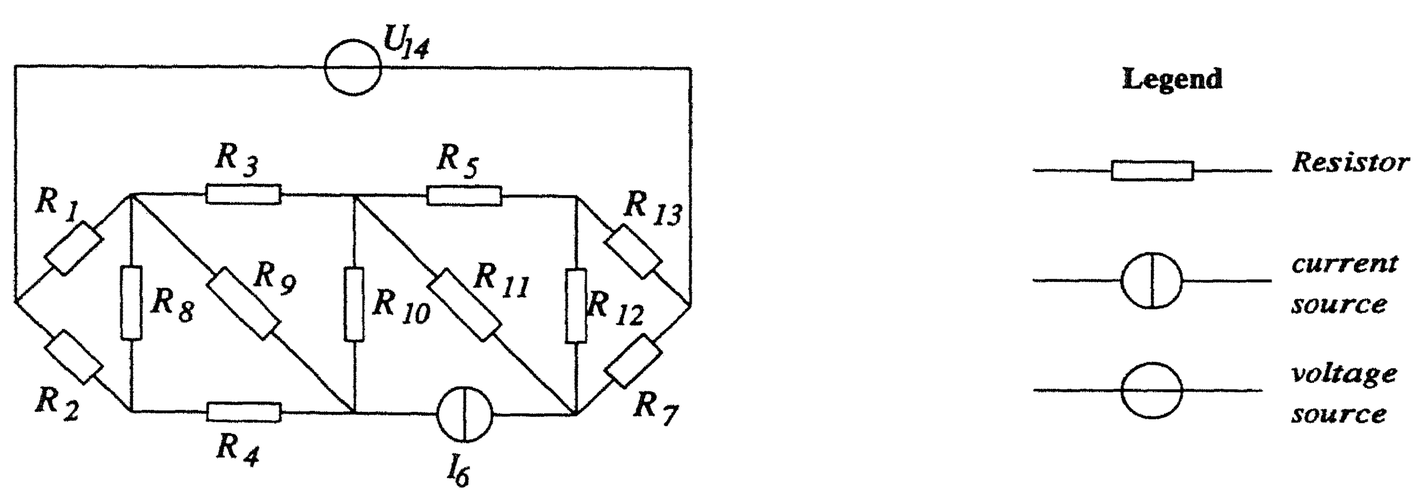


Figure 7.6: An electrical network.

is typically modeled by a *network graph* whose edges correspond to electric devises in which two edges are adjacent if the corresponding devices have a common terminal (see Figure 7.7).

Associated with each devise there are two oriented (time-depending) variables; its *potential difference* (or *voltage*) u and its *current* i. In the case

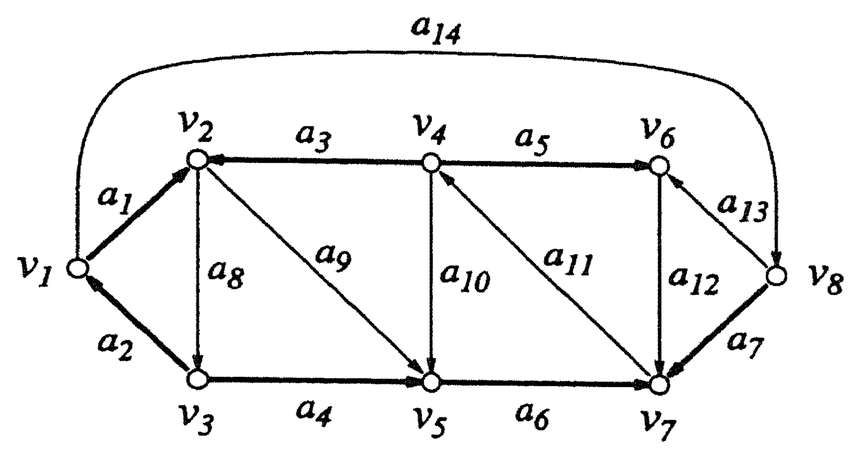


Figure 7.7: Network graph modeling the electrical network of Figure 7.6—the orientation of the edges is arbitrary.

of a voltage source, i is arbitrary and u is given. In the case of a current source, u is arbitrary and i is given. Finally, in the case of a resistor neither potential difference u nor current i are prescribed, instead they are assumed to be subject to *Ohm's law*: $u = Ri$, where R is a given nonnegative real number—the *resistance* of the resistor.

We use an arbitrary orientation of the edges of the network graph to orient both the potential difference between the terminals of a devise and the current through a devise. If an edge is oriented from vertex s to a vertex t then the potential difference is taken to be positive if the voltage on t is larger than the voltage on s while the current is taken as positive if it goes from s to t (i.e. it agrees with the orientation of the edge).

In analyzing the behavior of an electrical network one is interested in expressing the potential difference and current in each edge of network graph. Kirchhoff [1847] gave the necessary relation in order to fulfill this task:

Kirchhoff's voltage law: the sum of the potential differences around any circuit of the network graph (taking the orientations into account) is zero;

Kirchhoff's current law: the sum of currents along any cut of the network graph (taking the orientations into account) is zero. (Kirchhoff's original formulation can be seen to be equivalent to this one.)

For example, in (7.25) we show a maximal set of linearly independent Kirchhoff's voltage equations, i.e. the Kirchhoff's voltage equations corresponding to a basis of the circuit space of the digraph in Figure 7.7, where u_j denotes

the difference of potential between the endpoints of arc a_j $(j = 1, \ldots, n)$.

$$\begin{array}{lcl} u_1 + u_2 + u_8 & = & 0 \\ u_1 + u_2 - u_4 + u_9 & = & 0 \\ u_1 + u_2 - u_3 - u_4 + u_{10} & = & 0 \\ u_1 + u_2 - u_3 - u_4 - u_6 - u_{11} & = & 0 \\ u_1 + u_2 - u_3 - u_4 + u_5 - u_6 + u_{12} & = & 0 \\ u_1 + u_2 - u_3 - u_4 + u_5 - u_6 + u_7 - u_{13} & = & 0 \\ u_2 - u_4 - u_6 + u_7 + u_{14} & = & 0 \end{array} \tag{7.25}$$

In (7.26) we show a maximal set of linearly independent Kirchhoff's current equations, i.e. the Kirchhoff's current equations corresponding to a basis of the cut space of the digraph in Figure 7.7, where i_j denotes the current through arc a_j $(j = 1, \ldots, 14)$.

$$\begin{array}{lcl} i_1 - i_8 - i_9 - i_{10} + i_{11} - i_{12} + i_{13} & = & 0 \\ i_2 - i_8 - i_9 - i_{10} + i_{11} - i_{12} + i_{13} - i_{14} & = & 0 \\ i_3 + i_{10} - i_{11} + i_{12} - i_{13} & = & 0 \\ i_4 + i_9 + i_{10} - i_{11} + i_{12} - i_{13} + i_{14} & = & 0 \\ i_5 - i_{12} + i_{13} & = & 0 \\ i_6 - i_{11} + i_{12} - i_{13} + i_{14} & = & 0 \\ i_7 + i_{13} - i_{14} & = & 0 \end{array} \tag{7.26}$$

The above Kirchhoff's voltage and current equations plus Ohm's equations $u_j = R_j i_j$ for $j = 1, \ldots, 5, 7, \ldots, 13$ form a system with 14 equations in 14 unknowns (recall that i_6 and u_{14} are given). Together these systems can be seen to determine a unique solution—uniqueness is a very desirable property when designing electrical networks.

The basis of the circuit space used to write system (7.25) is formed by the fundamental circuits with respect to the spanning tree represented by thicker lines in Figure 7.7 and the basis of the cut space used to write the system (7.26) is formed by the fundamental cuts with respect to the spanning cotree represented by thinner lines in Figure 7.7. Had we chosen different basis of the circuit space and of the cut space to write down Kirchhoff's voltage and current equations then we could obtain, for instance, the systems (7.27) and (7.28) instead, respectively.

$$\begin{array}{lcl} u_1 + u_2 + u_8 & = & 0 \\ u_4 + u_8 - u_9 & = & 0 \\ u_3 + u_9 - u_{10} & = & 0 \\ u_6 + u_{10} + u_{11} & = & 0 \\ u_5 + u_{11} + u_{12} & = & 0 \\ u_7 - u_{12} - u_{13} & = & 0 \\ u_2 - u_4 - u_6 + u_7 + u_{14} & = & 0 \end{array} \tag{7.27}$$

$$\begin{array}{lcl} i_1 - i_2 + i_{14} & = & 0 \\ i_2 - i_8 + i_4 & = & 0 \\ i_3 + i_{10} - i_6 - i_{14} & = & 0 \\ i_4 - i_6 + i_9 + i_{10} - i_{11} & = & 0 \\ i_5 - i_{12} + i_{13} & = & 0 \\ i_3 - i_6 + i_{10} - i_{14} & = & 0 \\ i_7 + i_{13} - i_{14} & = & 0 \end{array} \tag{7.28}$$

Systems (7.25) and (7.27) are equivalent, in the sense that one can be obtained from the other by elementary row operations and, therefore, both have the same set of solutions. Yet (7.27) is simpler (sparser) than (7.25)! Similarly with respect to the systems (7.26) and (7.28).

Analyzing the discussion above one sees that the orientation of the edges of the network graph are irrelevant to the problem of finding sparse Kirchhoff's voltage equations and that this problem can be reduced to the to the shortest circuit basis problem (7.1).

Analogously, the problem of finding sparse Kirchhoff's current equations can be reduced to the *shortest cut basis problem* (i.e. the problem we obtain from problem (7.1) by replacing "circuit space" by "cut space"). We shall discuss the shortest cut basis problem in the next chapter.

Remark 7.1. The described application of shortest circuit bases and shortest cut bases to the so-called *cycle* and *cut analysis* of electrical networks is due to Chua & Chen [1973]. According to them, the saving in computation by considering shortest circuit bases and shortest cut bases may not be crucial for small linear electrical networks, but for nonlinear electrical networks (i.e. networks where each resistor is characterized by a noninvertible function $u = R(i)$) it is indeed decisive from both computational and theoretical point of view. For more graph theory applied to electrical networks see Bollobás [1979: Ch. 2], and Bryant [1967]. For an encyclopedic study see Recski [1989].

7.7. Further notes

We survey here some results related to (shortest) circuit bases.

Shortest bases of paths

Hartvigsen [1993] considered the following generalization of the shortest circuit basis problem. Let $G = (V, E)$ be an undirected graph. Let $U \subseteq V$ and $\mathcal{U} := \{U_1, \ldots, U_t\}$ be a partition of U. A *$\mathcal{U}$-path* in G is a path having both endpoints in U_i for some $i \in \{1, \ldots, t\}$. The *$\mathcal{U}$-space* of G is the linear subspace of $\mathrm{GF}(2)^{E(G)}$ generated by the characteristic vectors of circuits and $\mathcal{U}$-paths of G. A set of circuits and $\mathcal{U}$-paths whose incidence vectors form a basis for the

$\mathcal{U}$-space of G is called a $\mathcal{U}$-basis. The ***shortest $\mathcal{U}$-basis problem*** consists of:

$$\begin{array}{ll} \textit{given}: & -\text{ an undirected connected graph } G=(V,E); \\ & -\text{ a set } U \subseteq V \text{ and a partition } \mathcal{U} := \{U_1,\ldots,U_t\} \text{ of } U; \\ & -\text{ a length function } l : E \to \mathbb{R}_+; \\ \textit{find}: & -\text{ a } \mathcal{U}\text{-basis } \mathcal{B} \text{ of the } \mathcal{U}\text{-space of } G \text{ so that} \\ & \qquad \sum_{D\in\mathcal{B}} \sum_{e\in E(D)} l(e) \text{ is minimum.} \end{array} \tag{7.29}$$

Hartvigsen proposed a "Horton-type" algorithm to solve problem (7.29) (i.e., make a polynomially bounded list of circuits and $\mathcal{U}$-paths which are candidates for being a member of a shortest $\mathcal{U}$-basis and then extract the shortest $\mathcal{U}$-basis from this list with the greedy algorithm). Actually, Hartvigsen noted that the following collection of paths and cycles contains a shortest $\mathcal{U}$-basis of G[2] :

$$\begin{array}{ll} P_{r,v}\cdot e\cdot P_{w,r} & (r\in V;\ e=\{u,v\}\in E), \\ P_{u,v}\cdot e\cdot P_{w,u'} & (u,u'\in U_i\ (i=1,\ldots,t);\ e=\{v,w\}\in E), \end{array} \tag{7.30}$$

where $P_{x,y}$ denotes a shortest $x-y$-path in G. This algorithm finds a shortest $\mathcal{U}$-basis of a graph with n vertices and m edges in $O(n^2m^3)$ time.

The related problem of finding a shortest basis $\mathcal{B}$ of the $\mathcal{U}$-space of G (thus, each path in $\mathcal{B}$ need not be a $\mathcal{U}$-path) can be easily transformed to the shortest circuit basis problem; providing thus an $O(m^3+mn^2\log n)$-time algorithm to this problem. Indeed, let us be given an instance of the shortest U-basis problem, i.e. an undirected graph $G=(V,E)$, a set $U\subseteq V$, a partition $P=\{U_1,\ldots,U_t\}$ of U and a length function $l:E\to\mathbb{R}_+$. We denote by G' the graph obtained from G by identifying all vertices in U_i $(i=1,\ldots,t)$. So, for each vertex v of G there is a corresponding vertex v' of G' and if $u,v\in U_i$ for some $i\in\{1,\ldots,t\}$ then $u'=v'$. One can easily see that $\mathcal{B}$ is a shortest basis of the $\mathcal{U}$-space of G if and only if $\mathcal{B}$ is a shortest circuit basis of G'.

Minimum weight subgraph with a specified cyclomatic number

Hartvigsen [1993] also considered the problem of finding a minimum weight subgraph with a specified cyclomatic number (the *cyclomatic number* of a graph G is the dimension of its circuit space, denoted here by $\nu(G)$). More specifically,

[2]The similarity between (2.4), (7.7) and (7.30) is noteworthy.

Hartvigsen considered the following problem:

$$\begin{array}{ll} \textit{given}: & -\text{ an undirected connected graph } G=(V,E); \\ & -\ d \in \mathbb{Z}_+; \\ & -\text{ a weight function } w: E \to \mathbb{R}_+; \\ \textit{find}: & -\text{ a subgraph } G'=(V',E') \text{ of } G \text{ so that } \nu(G')=d \text{ and} \\ & \qquad \sum_{e\in E'} w(e) \text{ is minimum.} \end{array} \tag{7.31}$$

Extending his shortest $\mathcal{U}$-basis algorithm Hartvigsen presented an algorithm which solves problem (7.31) in time complexity $O(n^2m)$, when $d=1$, and time complexity $O(m^2n^{2d-1})$, when $d \geq 2$ and posed the following question:

> "Does there exist a polynomial time algorithm for finding a minimum weight subgraph whose cycle space has dimension k, where the polynomial does not depend on k being fixed?"

We show that problem (7.31) (stated as a decision problem) is $\mathcal{NP}$-complete. Thus, we give a negative answer to Hartvigsen's question (assuming $\mathcal{P} \neq \mathcal{NP}$).

A *clique* in a graph $G=(V,E)$ is a set of pairwise adjacent vertices of G. The *(maximum) clique problem* is the problem: *given:* a graph $G=(V,E)$ and a positive integer $k \leq |V|$; *question:* does G contain a clique of size at least k? The clique problem was shown to be $\mathcal{NP}$-complete by Karp [1972].

Theorem 7.13 *The problem (7.31) of finding a minimum weight subgraph with a specified cyclomatic number is $\mathcal{NP}$-complete.*

Proof. It is easy to see that problem (7.31) belongs to $\mathcal{NP}$. We transform the clique problem to problem (7.31) for instances where G is a simple graph and $w(e)=1$ for each edge e of G.

Let an instance of the clique problem be given, i.e. a simple graph $G=(V,E)$ and a positive integer $k \leq |V|$, $k \geq 3$. Corresponding to this instance we define $d := (k-1)(k-2)/2$ and $b := k(k-1)/2$. Now, we claim that G has a clique of size k if and only if there exists a subgraph $G'=(V',E')$ of G so that $\nu(G')=d$ and $|E'| \leq b$. In order to prove this claim we show that K_t is the unique simple graph (up to isomorphism) with cyclomatic number $(t-1)(t-2)/2$ and with at most $t(t-1)/2$ edges, $t \geq 3$.

Claim 7.8 *Let $H=(V,E)$ be a simple graph and $t \in \mathbb{Z}_+$, $t \geq 3$, so that $\nu(H)=(t-1)(t-2)/2$ and $|E| \leq t(t-1)/2$. Then H is isomorphic to K_t.*

Proof. Let $\omega(H)$ denote the number of components of H. We have that

$$\begin{aligned} \nu(H) &= (t-1)(t-2)/2 \\ &= |E| - (|V| - \omega(H)) \\ &\leq t(t-1)/2 - (|V| - \omega(H)). \end{aligned}$$

Thus, $|V| - \omega(H) \leq t-1$. Since H is a simple graph it follows that if $\omega(H) = 1$ then H is isomorphic to K_t and we are home. Now, we prove that H is indeed connected. Suppose to the contrary that $\omega(H) \geq 2$ and let $H_1 = (V_1, E_1), \ldots, H_{\omega(H)} = (V_{\omega(H)}, E_{\omega(H)})$ be the connected components of H. We may assume that $3 \leq |V_i| < t$ $(i = 1, \ldots, \omega(H))$. So we have that

$$\begin{aligned} \nu(H) &= (t-1)(t-2)/2 \\ &\leq \sum_{i=1}^{\omega(H)} ((|V_i| - 1)(|V_i| - 2)/2) \quad \text{(as } H \text{ is simple)} \\ &< (t-2) \sum_{i=1}^{\omega(H)} ((|V_i| - 1)/2), \end{aligned}$$

implying that $t - 1 < \sum_{i=1}^{\omega(H)} (|V_i| - 1)$. Therefore,

$$\begin{aligned} |E| &= \sum_{i=1}^{\omega(H)} (|V_i| - 1) + \nu(H) \\ &> t - 1 + \nu(H) \\ &= t(t-1)/2, \end{aligned}$$

a contradiction. ∎

A *k-cut* of a graph $G = (V, E)$ is a set $E' \subseteq E$ so that $G \setminus E'$ has at least k connected components. The *k-cut problem* is the problem: *given*: a graph $G = (V, E)$, a positive integer $k \leq |V|$ and a weight function $w : E \to \mathbb{R}_+$; *find*: a minimum weighted set of edges E' so that $G \setminus E'$ has at least k components. The k-cut problem was shown to be polynomially solvable for fixed k and $\mathcal{NP}$-complete in general (cf. Goldschmidt & Hochbaum [1994]). For planar graphs the k-cut problem and the problem (7.31) of finding a minimum weight subgraph with a specified cyclomatic number are dual problems. (If G is a planar graph embedded in $\mathbb{R}^2$ and G^* denotes its (geometric) dual then there is a one-to-one correspondence between the set of k-cuts of G and the subgraphs of G^* with cyclomatic number $k-1$.) A question in Hartvigsen's paper which remains unanswered is the following:

> "Does there exist a polynomial time algorithm for the planar k-cut problem, where the polynomial does not depend on k being fixed?"

A closely related problem is the *multiterminal cut problem*: *given:* a graph $G = (V, E)$, a set $T = \{t_1, \ldots, t_k\}$ of k terminal vertices and a weight

function $w : E \to \mathbb{R}_+$; *find:* a minimum weighted set $E' \subseteq E$ so that no pair of terminals belong to the same connected component of $G \setminus E'$. This problem is in a certain sense more difficult than the k-cut problem. The multiterminal cut problem is $\mathcal{NP}$-complete even for $k = 3$, and for planar graphs the multiterminal cut problem is $\mathcal{NP}$-complete for nonfixed k; on the positive side the problem is polynomially solvable for planar graph and fixed k (cf. Dahlhaus, Johnson, Papadimitriou, Seymour & Yannakakis [1994]).

Circuit bases and fundamental bases

Consider the graph in Figure 7.8 and the circuit basis formed by the three length-two circuits and one length-three circuit. This circuit basis is not fundamental. Hubicka & Sysło [1976] have given the following characterization of fundamental circuit bases.

Proposition 7.1 (Hubicka & Sysło [1976]) *A circuit basis $\mathcal{B}$ is fundamental if and only if each $C \in \mathcal{B}$ contains an edge that is in no $C' \in \mathcal{B}$, $C' \neq C$.* ∎

Figure 7.8 also shows that the length of a shortest fundamental circuit basis need not be equal to the length of a shortest circuit basis.

The edge lengths l for a graph $G = (V, E)$ is said *perturbed* if no two distinct subsets of edges E_1 and E_2 has the same sum of edge lengths (i.e. $E_1 \neq E_2$ implies $\sum_{e \in E_1} l(e) \neq \sum_{e \in E_2} l(e)$). Let $G = (V, E)$ be a graph with perturbed edge lengths and let us denote by C_e *the* shortest circuit containing the edge e. It is easy to prove that $\{C_e \mid e \in E\}$ is a set of independent circuits contained in *the* shortest circuit basis of G. Hartvigsen & Mardon [1993] have characterized those graphs for which $\{C_e \mid e \in E\}$ is the shortest circuit basis for every perturbed edge length. Figure 7.8 shows a graph such that $\{C_e \mid e \in E\}$ is *not* a circuit basis for each perturbed edge length.

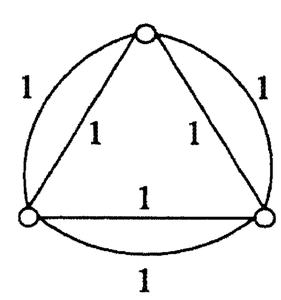


Figure 7.8: A graph for which *the* shortest circuit basis in not fundamental.

Hartvigsen & Zemel [1989] have characterized, by excluded minors, those graphs G for which the circuits of every circuit basis can be ordered in such a way that that $C_j \setminus (C_1 \cup \cdots \cup C_{j-1}) \neq \emptyset$ $(j = 2, \ldots, \nu)$. Fundamental circuit basis trivially satisfy this property: but are not the only ones, consider the graph of Figure 7.8. Hartvigsen & Mardon [1991] have defined classes of circuit bases of a graph based on the existence of an ordering $C_1, \ldots, C_\nu$ of its circuits

fulfilling certain property (as the property above: $C_j \setminus (C_1 \cup \cdots \cup C_{j-1}) \neq \emptyset$ $(j = 2, \ldots, \nu)$). For each one of these classes they have characterized those graphs for which every circuit basis is in a fixed class.

Further remarks

Ding [1994] has given a characterization of the minor-closed classes of graphs for which the number of circuits of each graph in the class is polynomially bounded by the number of edges of the graph. Thus, for graphs in those classes the enumeration approach solves in polynomial time, for instance, the shortest circuit basis problem and the Hamiltonian circuit problem.

For notes on shortest circuit bases in planar graphs see Section 6.4.

Chapter 8

Shortest bases of cuts

8.1. Introduction

In this chapter we focus on the following *shortest cut basis problem*:

given : – an undirected connected graph $G = (V, E)$;
– a length function $l : E \to \mathbb{R}_+$;

find : – a basis $\mathcal{B}^*$ of the cut space of G so that

$$\sum_{C \in \mathcal{B}^*} \sum_{e \in E(C)} l(e) \text{ is minimum.} \tag{8.1}$$

Our objective in the present chapter is twofold. Firstly, in Section 8.2, we observe that the algorithm dual (in the circuit-cut sense) to the one presented in Section 7.2 solves the shortest cut basis problem in polynomial time. As a by-product, in Section 8.3, we list results dual to the ones obtained in Section 7.3. Secondly, in Section 8.4, we observe that the polynomial-time algorithm developed by Gomory & Hu [1961] (see also Hu [1982: Ch. 2]) for solving the *all-pairs minimum cut problem* can be seen (via Gusfield's [1990] implementation of Gomory and Hu's method) as a particular realization of the algorithm presented in Section 8.2. Thus, Gomory and Hu's algorithm solves the shortest cut basis problem and conversely the algorithm described in Section 8.2 can easily be seen to solve the all-pairs minimum cut problem.

Before proceeding we observe here that the *shortest fundamental cut basis problem* (that is: *given:* an undirected connected graph G; *find:* a spanning cotree T^* of G so that the fundamental cut basis $\{C^*(T^*, e) \mid e \in E(G) \setminus E(T^*)\}$ is as short as possible) can be shown to be $\mathcal{NP}$-complete. Indeed, let T^* be a spanning cotree of $G = (V, E)$ and $T := G \setminus E(T^*)$. The length of the fundamental cut basis relative to T^* is

$$n - 1 + \sum_{\{u,v\} \in E(T^*)} |T_{u,v}|, \tag{8.2}$$

where $|T_{u,v}|$ denotes the length of the $u-v$-path in T. Finding a spanning cotree that minimizes (8.2) reduces to the problem of finding a spanning tree T of G that minimizes

$$\sum_{\{u,v\}\in E\setminus E(T)} |T_{u,v}|.$$

This latter problem is equivalent to the shortest fundamental circuit basis problem which was shown to be $\mathcal{NP}$-hard by Deo, Prabhu, Krishnamoorhty [1982]. We state this observation as the following theorem.

Theorem 8.1 *The shortest fundamental cut basis problem is $\mathcal{NP}$-complete.* ∎

Some additional notation and terminology

Let $G = (V, E)$ be an undirected connected graph and C^* be the set of cuts of G. We denote by $\nu^*(G)$ (or, simply ν^*) the dimension of the cut space of G. Thus, $\nu^* = n - 1$.

Let us be given a nonempty set T, $T \subseteq V$, with $|T|$ even. A *T-cut* is a set of edges of the form $\delta(W)$ with $|W \cap T|$ odd (the set T can be seen as a sign of the vertex set of G in the same way that Σ is a sign of the edge set of G). For a sign Σ of G let $T_\Sigma := \{v \in V \mid \text{the degree of } v \text{ in } G[\Sigma] \text{ is odd}\}$. It is easy to see that C^* is a cut of G with $|C^* \cap \Sigma|$ odd if and only if C^* is a T_Σ-cut.

A T-cut is a very well-known combinatorial object. However, a cut whose cardinality of the intersection with a sign Σ of G is odd is not a usual concept. Thus, in this chapter, we shall describe algorithms and theorems in the framework of T-cuts.

We say that cuts $C_1^*, \ldots, C_k^*$ of G are *independent* if their incidence vectors are linear independent in $\mathrm{GF}(2)^{E(G)}$. Analogously, we say that vertex sets $T_1, \ldots, T_k$ are *independent* if their incidence vectors are linear independent in $\mathrm{GF}(2)^{V(G)}$.

8.2. A polynomial-time algorithm

The minimum weight odd circuit problem plays a prominent role in our algorithm for the shortest circuit base problem (cf. Section 7.2). In case of the shortest cut basis problem that role will be taken over by the following *minimum weight T-cut problem*:

given : – an undirected graph $G = (V, E)$;
– a nonempty subset T of V with $|T|$ even;
– a weight function $w : E \to \mathbb{R}_+$;

find : – a T-cut $\delta(W)$ in G so that (8.3)

$$\sum_{e\in\delta(W)} w(e) \text{ is minimum.}$$

Padberg & Rao [1982] have shown that the minimum weighted T-cut problem is polynomially solvable. In particular, when $|T| = 2$, problem (8.3) is the *minimum (weighted) $\{r, s\}$-cut problem* (where $\{r, s\} = T$); so it can be solved by any algorithm for the maximum flow minimum cut problem (see, for instance, Tarjan [1983]).

The algorithm—a dual version of the algorithm in Section 7.2

Let us be given an undirected connected graph $G = (V, E)$ and a nonnegative length function $l : E \to \mathbb{R}_+$. We find a shortest cut basis of G as follows:

Step 1. Let $\{e_1, \ldots, e_{\nu^*}\}$ be the edge set of a spanning tree of G.
Define $T_{1,i} := e_i$ $(i = 1, \ldots, \nu^*)$.

Step 2. For $k := 1, \ldots, \nu^*$ iterate the following:

Find a minimum weighted $T_{k,k}$-cut C_k^* in G (with respect to the weight function l);

Define for $i = k+1, \ldots, \nu^*$:

$$T_{k+1,i} := \begin{cases} T_{k,i} \,\Delta\, T_{k,k} & \text{if } C_k^* \text{ is } T_{k,i}\text{-cut;} \\ T_{k,i} & \text{otherwise.} \end{cases}$$

The algorithm returns $\{C_1^*, \ldots, C_{\nu^*}^*\}$.

Correctness of the algorithm

The correctness of the algorithm above can be proved in a way identical to the proof of the correctness of the algorithm in Section 5.2—as one would expect. Thus, we enunciate the corresponding lemmas but omit their proofs. We observe that $T_{k,k} \neq \emptyset$ as $T_{k,k}$ always contains at least one of the endpoints of e_k $(k = 1, \ldots, \nu^*)$.

Lemma 8.1 *Let C^* be a cut of G so that C^* is a T-cut and a T'-cut (C^* is neither a T-cut nor T'-cut). Then C^* is* **not** *a $T \,\Delta\, T'$-cut.* ∎

Lemma 8.2 *Let $C_1^*, \ldots, C_k^*$ be independent cuts and $\emptyset \neq T \subseteq V$ with $|T|$ even so that C_i^* is* **not** *a T-cut $(i = 1, \ldots, k)$. Then $C_1^*, \ldots, C_k^*, C^*$ are independent cuts for any T-cut C^*.* ∎

Lemma 8.3 *Let C_k^* be the $T_{k,k}$-cut found during iteration k of Step 2 $(k = 1, \ldots, \nu^* - 1)$. Then C_k^* is* **not** *a $T_{p,i}$-cut $(k + 1 \leq p \leq i \leq \nu)$.* ∎

Theorem 8.2 *The set $\{C_1^*, \ldots, C_{\nu^*}^*\}$ returned by the algorithm is a shortest cut basis of G.* ∎

Complexity of the algorithm

Padberg & Rao [1982] showed that the minimum weighted T-cut problem can be solved by ν^* applications of an algorithm for the minimum weighted $\{r,s\}$-cut problem (i.e, for the case when $|T| = 2$). This latter problem can be solved in $O(\min\{nm\log n\ ,\ n^3\})$ time (cf. Sleator & Tarjan [1983] and Karzanov [1974]—see also Ahuja, Magnanti & Orlin [1993: Ch. 8]). Thus, the total time needed by Step 2 to find a minimum weighted $T_{k,k}$-cut ($k = 1,\ldots,\nu^*$) is $O((\nu^*)^2 \min\{nm\log n\ ,\ n^3\})$. The total time required by the algorithm for updating the sets $T_{i,j}$ is upper bounded by $O((\nu^*)^3)$. Therefore, the resulting running time of the algorithm presented is $O((\nu^*)^3+(\nu^*)^2 \min\{nm\log n\ ,\ n^3\}) = O(\min\{n^3m\log n\ ,\ n^5\})$. (In the next section we shall see that it is possible to realize the algorithm in such a way that $|T_{i,j}| = 2$ ($1 \le i \le j \le \nu^*$). This speeds up the running time by a factor of n.) We have established the following result.

Theorem 8.3 *Solving the shortest cut basis problem (8.1) requires $O((\nu^*)^3)$ time plus the time required to solve ν^* instances of the minimum weighted T-cut problem.* ∎

8.3. Bases of cuts and signed graphs

Dual results to the ones obtained in Section 7.3 hold here as well. We state, without proof, the dual version of Theorems 7.3, 7.4 and 7.5.

Theorem 8.4 *The length of a shortest cut basis of G is equal to*

$$\max_{T_1,\ldots,T_{\nu^*}} \sum_{i=1}^{\nu^*} \{l(C_i^*) \mid C_i^* \text{ is a minimum weighted } T_i\text{-cut}\},$$

where the maximum ranges over all independent vertex sets $T_1,\ldots,T_{\nu^*}$ of G with $|T_i|$ even ($i = 1,\ldots,\nu^*$). ∎

Let us remark that statements analogous to the two following theorems also hold if we replace shortest cut basis and minimum weighted T-cut by cut basis and T-cut, respectively.

Theorem 8.5 *Let $\mathcal{B}^*$ be a set of cuts of G. Then the following are equivalent:*

(*i*) *$\mathcal{B}^*$ is a shortest cut basis of G;*
(*ii*) *$\mathcal{B}^*$ is a minimal set of cuts of G so that $\mathcal{B}^*$ contains a minimum weighted T-cut for every vertex set T of G with $|T|$ even.* ∎

Theorem 8.6 *Let $\mathcal{B}^* := \{C_1^*, \ldots, C_{\nu^*}^*\}$ be a set of cuts of G. Then the following are equivalent:*

(i) $\mathcal{B}^*$ is a shortest cut basis of G;
(ii) there exist vertex sets $T_1, \ldots, T_{\nu^*}$ with $|T_i|$ even $(i = 1, \ldots, \nu^*)$ so that the following holds for $k = 1, \ldots, \nu^*$:
(a) C_k^* is a minimum weighted T_k-cut in G; and
(b) C_k^* is not a T_i-cut in G $(i = 1, \ldots, \nu^*;\ i \neq k)$.

∎

8.4. Shortest bases of cuts and the all-pairs minimum cut problem

Let $G = (V, E)$ be an undirected connected graph and $w : E \to \mathbb{R}_+$ a weight function on its edges. The *all-pairs minimum cut problem* requires us to determine the value $\mathrm{val}_{G,w}(u, v)$ of a minimum weighted $\{u, v\}$-cut for every pair of vertices $\{u, v\}$ of G. This problem can be trivially solved by $n(n-1)/2$ (maximum flow) minimum cut computations. Gomory & Hu [1961] have shown that $n - 1$ computations suffice. Furthermore, they have shown that there exists always a tree $T = (V, F)$ and a weight function $w_T : F \to \mathbb{R}_+$ so that

$$\mathrm{val}_{G,w}(u, v) = \mathrm{val}_{T,w_T}(u, v) \quad \text{for every pair of vertices } \{u, v\} \text{ of } G. \tag{8.4}$$

A tree T satisfying the above properties is called a *flow-equivalent tree* (relative to G and w). (Note that, in general, F is not a subset of E.) Gomory & Hu have actually shown a stronger result. We say that a tree T is a *Gomory-Hu tree* (or a *cut-equivalent tree*) for G if T is a flow-equivalent tree so that for each edge $e = \{u, v\}$ of T, the cut C_e^* of G formed by those edges of G connecting the two components of $T - e$ is a minimum weighted $\{u, v\}$-cut. Figure 8.1 shows a graph G, a flow-equivalent tree and a Gomory-Hu tree for G.

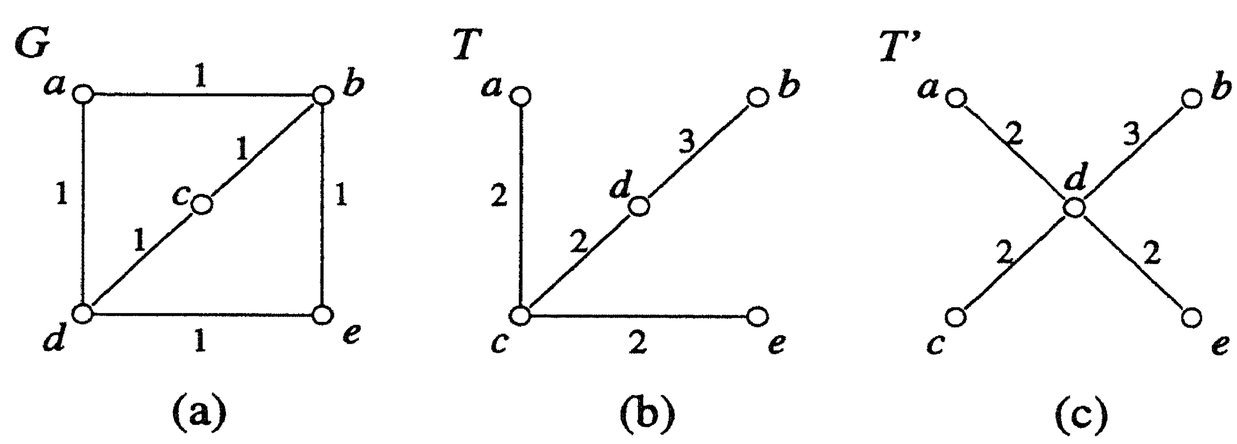


Figure 8.1: (a) shows a graph G with weights on its edges; (b) a flow-equivalent tree T for G; and (c) a Gomory-Hu tree T' for G.

Gomory & Hu [1961] have shown the following theorem (see also Hu [1982: Ch. 2], Lovász & Plummer [1986: Ch. 2] and Ahuja, Magnanti & Orlin [1993: Ch. 8]).

Theorem 8.7 (The Cut-equivalent Tree Theorem)

Each undirected graph with nonnegative weights on its edges has a Gomory-Hu tree. ∎

Let $G = (V, E)$ be a graph. We say that two subsets X and Y of V are *cross-free* if

$$X \subseteq Y, \quad Y \subseteq X, \quad Y \cap X = \emptyset \quad \text{or} \quad X \cup Y = V.$$

Two cocycles $\delta(X)$ and $\delta(Y)$ are *cross-free* if X and Y are cross-free.

For each edge e of a Gomory-Hu tree T let X_e and $V \setminus X_e$ be the vertex set of the two components of $T - e$. If $\{e_1, \ldots, e_{\nu^*}\}$ is the edge set of a Gomory-Hu tree then we call the cut set $\{\delta(X_{e_1}), \ldots, \delta(X_{e_{\nu^*}})\}$ a *Gomory-Hu family of cuts.*

Gomory-Hu family of cuts and shortest bases of cuts

Let $G = (V, E)$ be a graph and $l : E \to \mathbb{R}_+$. We establish here the relationship between Gomory-Hu family of cuts of G and shortest cut bases of G.

Lemma 8.4 *Every Gomory-Hu family of cuts is a (cross-free) shortest cut basis.*

The proof of Lemma 8.4 is analogous to the proof of Theorem 7.1. This lemma combined with the Cut-equivalent Tree Theorem (Theorem 8.7) implies that each undirected graph has a shortest cut basis whose cuts are pairwise cross-free, that is, a *cross-free shortest cut basis.*

Proof of Lemma 8.4. Let T be a Gomory-Hu tree and let $\mathcal{B}^*$ be the corresponding Gomory-Hu family of cuts. Consider an edge $e = \{u, v\} \in E(T)$. The cut $\delta(X_e)$ is the unique $\{u, v\}$-cut in $\mathcal{B}^*$ (recall that X_e denotes the vertex set of an arbitrary component of $T - e$). By induction on the components of $T - e$ and by Lemma 8.2 it follows that $\mathcal{B}^*$ is a cut basis of G.

Suppose by contradiction that $\mathcal{B}^*$ is not a shortest cut basis and let $\mathcal{B}'$ a shortest cut basis of G so that $|\mathcal{B}^* \cap \mathcal{B}'|$ is maximum. Let $e \in E(T)$ so that $C_e^* := \delta(X_e) \in \mathcal{B}^* \setminus \mathcal{B}$. Since $\mathcal{B}'$ is a cut basis we have that

$$C_e^* \;=\; C_1' \,\Delta\, C_2' \,\Delta\, \cdots \,\Delta\, C_t',$$

for distinct cuts $C_1', \ldots, C_t'$ in $\mathcal{B}'$. Let u and v be the endpoints of e. There must exist a $k \in \{1, \ldots, t\}$ so that C_k' is an $\{u, v\}$-cut implying that $l(C_e^*) \leq l(C_k')$. Hence, $\mathcal{B}'' := (\mathcal{B}' \setminus \{C_k'\}) \cup \{C_e^*\}$ is a shortest cut basis of G so that $|\mathcal{B}'' \cap \mathcal{B}^*| > |\mathcal{B}' \cap \mathcal{B}^*|$, a contradiction. ∎

Remark 8.1. Let $\mathcal{B}^*$ be a shortest cut basis of G. From the proof of Lemma 8.4 (or by Theorem 8.5) it is easy to see that $\mathcal{B}^*$ contains a minimum weighted $\{u, v\}$-cut for each pair $\{u, v\}$ of vertices of G. Hence, in particular, a shortest cut basis provides a solution to the all pairs minimum cut problem.

Lemma 8.5 *Every cross-free shortest cut basis is a Gomory-Hu family of cuts.*

Proof. Let $\mathcal{B}^* := \{\delta(X_1), \ldots, \delta(X_{\nu^*})\}$ be a cross-free shortest cut basis. We have to show that there exists a tree $T = (V, F)$ and a weight function $w_T : F \to \mathbb{R}_+$ satisfying (8.4) so that T is a Gomory-Hu tree (relative to l) having $\mathcal{B}^*$ as Gomory-Hu family of cuts.

We may assume that not all cuts in $\mathcal{B}^*$ are stars. Indeed, suppose that $V = \{v_1, \ldots, v_{\nu^*+1}\}$ and $\delta(X_i) = \delta(\{v_i\})$ $(i = 1, \ldots, \nu^*)$. Let $F := \{\{v_{\nu^*+1}, v_i\} \mid i = 1, \ldots, \nu^*\}$. The star $T = (V, F)$ is a Gomory-Hu tree having $\mathcal{B}^*$ as Gomory-Hu family of cuts and we are done..

Hence, for some distinct vertices u and v of G we may assume that there exists a t, $1 \leq t \leq \nu^*$ so that the following holds:

(*i*) $\delta(X_t)$ is not a star;
(*ii*) $\delta(X_t)$ is a minimum weighted $\{u, v\}$-cut; and
(*iii*) $X_i \subset X_t$ $(i = 1, \ldots, t-1)$ and $X_i \subset V \setminus X_t$ $(i = t+1, \ldots, \nu^*)$.

Hence, $\mathcal{B}_1 := \{\delta(X_1), \ldots, \delta(X_t)\}$ and $\mathcal{B}_2 := \{\delta(X_t), \ldots, \delta(X_{\nu^*})\}$ are cross-free shortest cut basis of $G_1 := G/(V \setminus X_t)$ and of $G_2 := G/X_t$, respectively. By induction, there exist $T_i = (V_i, F_i)$ and $w_{T_i} : F_i \to \mathbb{R}_+$ so that T_i is a Gomory-Hu tree of G_i having $\mathcal{B}_i$ as Gomory-Hu family of cuts $(i = 1, 2)$. Thus, the tree $T = (V, F_1 \cup F_2 \cup \{u, v\})$ with edge weights w_T defined as:

$$w_T(f) := \begin{cases} w_{T_i}(f) & \text{if } f \in F_i \ (i = 1, 2); \\ l(\delta(X_t)) & \text{if } f = \{u, v\} \end{cases}$$

is a Gomory-Hu tree of G having $\mathcal{B}^*$ as Gomory-Hu family of cuts. ∎

The following theorem and corollary can be seen as a sharpening of Theorems 8.5 and 8.6 for cross-free cut bases and as a generalization of a theorem in Hartvigsen & Mardon [1994].

Theorem 8.8 *Let G be a graph with nonnegative lengths on its edges and let $\mathcal{B}^*$ be a set of cross-free cuts of G. Then the following are equivalent:*

(*i*) *$\mathcal{B}^*$ is a Gomory-Hu family of cuts;*
(*ii*) *$\mathcal{B}^*$ is a cross-free shortest cut basis; and*
(*iii*) *$\mathcal{B}^*$ is a minimal set of cuts of G so that $\mathcal{B}^*$ contains a minimum weighted $\{u,v\}$-cut for every pair of vertices $\{u,v\}$ of G.*

Proof. By Lemmas 8.4 and 8.5, (*i*) $\Leftrightarrow$ (*ii*), and, by Theorem 8.5 (*ii*) $\Rightarrow$ (*iii*). So remains to prove (*iii*) $\Rightarrow$ (*ii*).

Let $\mathcal{B}^* := \{\delta(X_1), \ldots, \delta(X_t)\}$. From the minimality of $\mathcal{B}^*$ it follows that $\{\delta(X_1), \ldots, \delta(X_t)\}$ are independent cuts. We may assume that $X_1, \ldots, X_t$ are

cross-free. Thus, by a result due to Edmonds & Giles [1977] there exist a directed tree $T = (W, A)$ with $A = \{1, \ldots, t\}$ and a function $\phi : V \to W$ so that for $i = 1, \ldots, t$ the following holds:

$$X_i = \{v \in V \mid \phi(v) \text{ is in the same component of the graph } (W, A \setminus \{i\}) \text{ as the head of the arc } i\}.$$

(The pair T, ϕ is said a *tree-representation* for the cross-free family $\{X_1, \ldots, X_t\}$ (see Schrijver [1986: p. 278]).) We claim that ϕ is injective. Indeed, if there exist distinct vertices u and v of G so that $\phi(u) = \phi(v)$ then either $\{u, v\} \subseteq X_i$ or $\{u, v\} \subseteq V \setminus X_i$ $(i = 1, \ldots, t)$ implying that $\mathcal{B}^*$ does not contain a $\{u, v\}$-cut, a contradiction. Thus $t = \nu^*$ and $\mathcal{B}^*$ is a cut basis of G. Now, analogously to the proof of Lemma 8.5, one can constructed from $\mathcal{B}^*$ a Gomory-Hu tree having $\mathcal{B}^*$ as Gomory-Hu family of cuts, implying, by Lemma 8.4, that $\mathcal{B}^*$ is a cross-free shortest cut basis. ∎

Corollary 8.1 *Let G be a graph with nonnegative lengths on its edges and let $\mathcal{B}^* := \{C_1^*, \ldots, C_{\nu^*}^*\}$ be a set of cuts of G. Then the following are equivalent:*

- *(i) $\mathcal{B}^*$ is a cross-free shortest cut basis of G;*
- *(ii) there exist vertex sets $\{u_1, v_1\}, \ldots, \{u_{\nu^*}, v_{\nu^*}\}$ so that the following holds for $k = 1, \ldots, \nu^*$:*
  - *(a) C_k^* is a minimum weighted $\{u_k, v_k\}$-cut in G; and*
  - *(b) C_k^* is not a $\{u_i, v_i\}$-cut in G $(i = 1, \ldots, \nu^*;\ i \neq k)$.*

Proof. Take as $\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_{\nu^*}, v_{\nu^*}\}$ the edge set of a Gomory-Hu tree and apply the equivalence in Theorem 8.8. ∎

Remark 8.2. Corollary 8.1 implies that each T_i in the enunciate of Theorem 8.4 can be restricted to a vertex pair. In particular, this implies that the shortest fundamental cut basis problem is polynomially solvable for complete graphs.

A faster shortest cut basis algorithm—Gusfield's method

Gusfield [1990] had the following motivation to design a new implementation of Gomory-Hu's method for finding a Gomory-Hu tree:

> "It is easy to underestimate the amount of programming detail needed by the original method of Gomory and Hu. In fact, the ideas leading to this paper partly began after a failed attempt to quickly implement the method. The implementation was made more difficult because we used existing code for finding the maximum flow, but we did not understand the code well, and we needed to modify it to implement graph contraction and expansion. ..."

Gusfield's algorithm (and therefore Gomory-Hu's algorithm as well) can be seen as a realization of the algorithm in Section 8.2 that fulfills $|T_{i,j}| = 2$

$(1 \leq i \leq j \leq \nu^*)$. Gusfield's algorithm is as follows. Given are an undirected connected graph $G = (V, E)$ and a function $l : E \to \mathbb{R}_+$.

Step 1. Let $\{v_1, v_2, \ldots, v_n\}$ be the vertex set of G.
Define $T_{1,i} := \{v_i, v_n\}$ $(i = 1, \ldots, \nu^*)$. ($T_{1,i}$ is not necessarily an edge of G $(i = 1, \ldots, \nu^*)$.)

Step 2. For $k := 1, \ldots, \nu^*$ iterate the following:

Find a minimum weighted $T_{k,k}$-cut C_k^* in G (with respect to the weight function l);

Define for $i = 1, \ldots, \nu^*$:

$$T_{k+1,i} := \begin{cases} T_{k,i} \,\Delta\, T_{k,k} & \text{if } C_k^* \text{ is } T_{k,i}\text{-cut and} \\ & |T_{k,i} \cap T_{k,k}| = 1\ ; \\ T_{k,i} & \text{otherwise.} \end{cases}$$

The algorithm returns the Gomory-Hu tree H_{ν^*} formed by the edge set $\{T_{\nu^*,1}, T_{\nu^*,2}, \ldots, T_{\nu^*,\nu^*}\}$, where $w_H(T_{\nu^*,k}) = l(C_k^*)$ $(k = 1, \ldots, \nu^*)$.

Figure 8.2 illustrates the application of the Gusfield's algorithm in the graph in Figure 8.2(a). Note that the cut set $\{C_1^*, \ldots, C_5^*\}$ computed by the algorithm does not form a cut basis. We do need the returned Gomory-Hu tree in order to get the desired shortest cut basis.

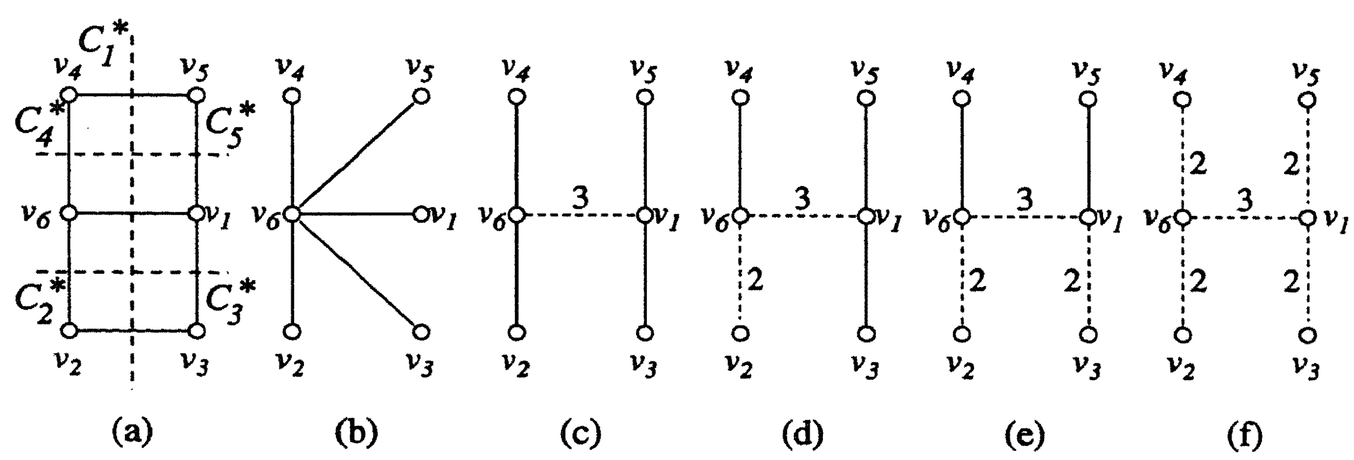


Figure 8.2: Illustration of Gusfield's algorithm. (a) shows the graph G and the cuts found by the algorithm. Figures (b), (c), (d), (e), (f) shows the tree formed by the edge set $\{T_{k,1}, \ldots, T_{k,5}\}$ for $k = 1, 2, 3, 4$ and 6, respectively. The dotted lines represent the edges $\{T_{k,i} \mid k < i\}$.

The correctness of the algorithm follows from the following theorem.

Theorem 8.9 *For $k = 1, \ldots, \nu^*$ the following holds:*

(i) $\{T_{k,1}, T_{k,2}, \ldots, T_{k,\nu^*}\}$ is the edge set of a tree H_k;

(*ii*) *each component of* $H'_k := H_k \setminus \{T_{k,1}, \ldots, T_{k,k-1}\}$ *is a* star *(i.e. isomorphic to* $K_{1,t}$ *for some* $t \in \mathbb{Z}_+$*);*

(*iii*) *the endpoints of* $T_{k,i}$ *are the centers of stars in* H'_k $(i = 1, \ldots, k-1)$ *(by the* center of a star *we mean a vertex of maximum degree in the star);*

(*iv*) *those edges of* G *connecting the two components of* $H_k - T_{k,i}$ *is a minimum weighted* $T_{k,i}$*-cut* $(i = 1, \ldots, k-1)$.

Using solely the definition of $T_{i,j}$ $(i = 1, \ldots, \nu^*;\ j = 1, \ldots, \nu^*)$ one can easily prove Theorem 8.9 (*i*), (*ii*) and (*iii*) by induction on k. The proof of (*iv*) is based on the following well-known "uncrossing procedure".

Proposition 8.1 (Gomory & Hu [1961]) *Let* r, x *and* y *be vertices of a graph* $G = (V, E)$ *with nonnegative weights on its edges and let* $\delta(X)$ *and* $\delta(Y)$ *be a minimum weighted* $\{r, x\}$*-cut and a minimum weighted* $\{r, y\}$*-cut, respectively, so that* $x \in X$ *and* $y \in Y$*. Then*

(*i*) $y \in V \setminus X$ *implies that* $\delta(X \setminus Y)$ *is a minimum weighted* $\{r, x\}$*-cut;*

(*ii*) $y \in X$ *implies that* $\delta(X \cup Y)$ *is a minimum weighted* $\{r, x\}$*-cut.* ∎

Proof of Theorem 8.9 (*iv*). The proof is by induction on k. Note that (*iv*) trivially holds for $k = 1, 2$. Let us suppose that (*iv*) is true for $k-1$, $k \geq 3$. Let $T_{k-1,k-1} = \{r, x\}$, and $\{r, y_1\}, \{r, y_2\}, \ldots, \{r, y_{t_r}\}$ be the edges in $\{T_{k-1,1}, \ldots, T_{k-1,k-2}\}$ having r as one of its endpoints; we may assume that $T_{k-1,i} = \{r, y_i\}$ $(i = 1, \ldots, t_r)$. (Thus, we are assuming that r is the center of the star of H'_{k-1} containing the edge $\{r, s\}$). Moreover, let $C^*_k = \delta(X)$ be the minimum weighted $\{r, x\}$-cut found during the kth iteration of Step 2 and Y_i be the component of $H_{k-1} - \{r, y_i\}$ containing y_i $(i = 1, \ldots, t_r)$ (by induction, $\delta(Y_i)$ is a minimum weighted $\{r, y_i\}$-cut). Figure 8.3 illustrates the definitions and the proof.

By definition $T_{k,i} = T_{k-1,i}$ $(i = t_r + 1, t_r + 2, \ldots, k-2)$ and the vertex set of each component of $H_{k-1} - T_{k-1,i}$ is the vertex set of a component of $H_k - T_{k,i}$. Hence, for these edges (*iv*) holds trivially. Also by definition, the edge $T_{k,i}$ $(i = 1, \ldots, t_r)$ connects y_i either to r or to x. In either cases it is easy to verify that Y_i is the vertex set a component of $H_k - T_{k,i}$. Thus, also for these edges (*iv*) holds. It remains to show that (*iv*) holds for $T_{k,k-1} = \{r, x\}$. Consider the vertex set X. By Proposition 8.1 we may assume that $Y_i \subset X$ if $y_i \in X$ and that $X \cap Y_i = \emptyset$ if $y_i \in V \setminus X$ $(i = 1, \ldots, t_r)$. This shows that $\delta(X)$ is the edge set of G connecting the two components of $H_k - \{r, x\}$ (note that $V = Y_1 \cup \ldots \cup Y_{t_r} \cup S$ where S is the vertex set of the star of H'_{k-1} containing the edge $\{r, s\}$). ∎

Comparing the algorithm in Section 8.2 with Gusfield's algorithm one sees quite a few differences. To see that Gusfield's algorithm is a version of the algorithm is Section 8.2 one should realize that with the $T_{k,k}$-cut $\delta(X)$

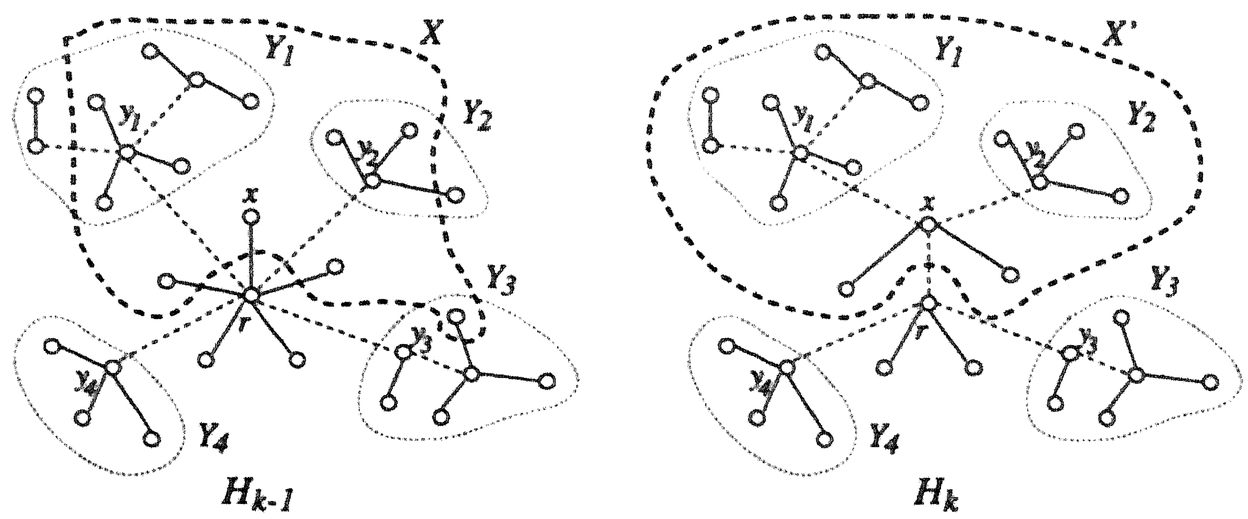


Figure 8.3: H_p denotes the tree formed by the edge set $\{T_{p,1}, \ldots, T_{p,\nu^*}\}$; the edges $T_{p,1}, \ldots, T_{p,p-1}$ are represented in the figure by dashed lines. H'_p is the forest formed by edges represented by solid lines. In the figure r, y_1, y_2, y_3 and y_4 are centers of stars in H'_{k-1} and $\delta(X)$ and $\delta(X')$ are minimum weighted $\{r, x\}$-cuts in G.

explicitely calculated in Gusfield's algorith one implicitely also has a $T_{k,k}$-cut $\delta(X')$ like in Figure 8.3 and if the T's would be updated according to the rule in Section 8.2 using that implicit cut one gets Gusfiefd's rule. Other difference between the algorithms is that Gusfield's algorithm updates all T's; the reason for that is to remember the implicit cuts.

As a consequence of the algorithms of Gomory & Hu [1961] and Gusfield [1990] combined with Theorem 8.8 we have the following:

Theorem 8.10 *Solving the shortest cut basis problem (8.1) requires $O((\nu^*)^2)$ time plus the time required to solve ν^* instances of the minimum $\{r, s\}$-cut problem.* ∎

Theorem 8.10 implies that the shortest cut basis problem can be solved in

$$O(\min\{n^2 m \log n \ , \ n^4\}) \ \text{ time.}$$

(The maximum flow minimum cut algorithms due to Sleator & Tarjan [1983] and Karzanov [1974] run in $O(nm \log n)$ and $O(n^3)$ time, respectively (see also Ahuja, Magnanti & Orlin [1993: Ch. 8]).)

8.5. Further notes: shortest cut and circuit bases in planar graphs

Let $G = (V, E)$ be a simple connected planar graph embedded in $\mathbb{R}^2$ and $l : E \to \mathbb{R}_+$. We denote by $G^* = (\mathcal{F}, E^*)$ the (*geometric*) *dual* of G and by e^* the dual edge of e for all $e \in E$. Note that $\nu(G) = \nu^*(G^*)$. Let $l^* : E^* \to \mathbb{R}_+$ so that $l^*(e^*) = l(e)$ for all $e \in E$. It can easily be seen that $\{C_1, \ldots, C_\nu\}$ forms a shortest circuit basis of G (with respect to the edge length l) if and

only if $\{C_1^*, \ldots, C_\nu^*\}$ forms a shortest cut basis of G^* (with respect to the edge length l^*), where $C_i^* = \{e^* \mid e \in C_i\}$ $(i = 1, \ldots, \nu)$.

Itai & Shiloach [1979] have shown that the minimum weighted $\{r, s\}$-cut problem in planar graphs can be solved in $O(n^2 \log n)$ time. Reif [1983], using a divide-and-conquer approach, described a more efficient implementation of Itai & Shiloach's algorithm (see also Hassin & Johnson [1985]). This implementation runs in $O(n(\log n)^2)$ time. All these algorithms use as subroutine Johnson's implementation of Dijkstra's shortest path algorithm (cf. Johnson [1977]). Frederickson [1987] provided further improvements and showed that Reif's algorithm can run in $O(n \log n)$ time if Frederickson's all-pairs shortest path algorithm is used. Reif's and Frederickson's algorithms combined with Gomory-Hu's method leads to an algorithm that finds a shortest cut basis of a planar graph in $O(n^2 \log n)$ time (see Theorem 8.10). This is the current best running time for solving the shortest cut (circuit) basis problem in planar graphs.

Both Frederickson's and Gomory & Hu's algorithm (especially the former) require the use of fairly sophisticated data structures. This served as motivation to Hartvigsen & Mardon [1994] developing an $O(n^2 \log n)$-time algorithm for the shortest circuit basis problem in planar graphs which uses none of these algorithms. They have modified Horton's algorithm (cf. Horton [1987]; see Section 5.2) in order to shorten the list of circuits and then extracted the shortest circuit basis from this list without using Gaussian elimination (the Gaussian elimination alone would take $O(n^3)$ time). (The algorithm presented in Section 5.2 runs in $O(n^3 \log n)$ time in planar graphs.)

Chapter 9

Shortest bases of circuits in binary matroids

9.1. Introduction

The graph-theoretic problems investigated in Chapter 7 and 8 can be generalized naturally to binary matroids. A *binary matroid* M on a finite ground set E consists of a linear subspace $\mathcal{C}(M)$ of $\mathrm{GF}(2)^E$. $\mathcal{C}(M)$ is called the *circuit space* of M. Set-theoretically a binary matroid M on E is a set $\mathcal{C}(M)$ of subsets of E closed under taking symmetric differences. An element of $\mathcal{C}(M)$ is called a *cycle* of M and a inclusion-wise minimal non-empty cycle is called a *circuit*.

The problem we consider in the present chapter is the following *shortest circuit basis problem*:

$$\begin{array}{ll} \textit{given}: & \text{– a binary matroid } M \text{ on } E; \\ & \text{– a length function } l : E \to \mathbb{R}_+; \\ \textit{find}: & \text{– a basis } \mathcal{B} \text{ of the circuit space of } M \text{ so that} \\ & \displaystyle\sum_{C \in \mathcal{B}} l(C) \text{ is minimum.} \end{array} \tag{9.1}$$

Here we are assuming that a binary matroid is given by a $\{0,1\}$-matrix $A := [I \mid B]$ where E indexes the columns of A, I denotes the identity matrix of appropriate dimension and $\mathcal{C}(M)$ is the *nullspace* of A (*i.e.*, $\mathcal{C}(M) = \{x \in \mathrm{GF}(2)^E \mid Ax = 0\}$). Matrix A is called a *(standard) representation matrix* of M. A solution to problem (9.1) is said an *shortest circuit basis*.

Problem (9.1) is a common generalization of both the shortest circuit basis problem and shortest cut basis problem. A binary matroid M is said to be *graphic* (*cographic*) if $\mathcal{C}(M)$ is isomorphic to the circuit space (cut space, respectively) of a graph.

So, in Chapters 7 and 8 we have seen that problem (9.1) is polynomially solvable for the classes of graphic and cographic matroids, respectively. Our objective in this chapter is showing that the shortest circuit basis problem is polynomially solvable for a larger class of binary matroids, namely the binary matroids with no S_8-minor (this class is defined in Section 9.2).

Before proceeding we observe here that problem (9.1) is in general $\mathcal{NP}$-complete. D.S. Johnson showed that the following *shortest f-circuit problem* (stated as a decision problem) is $\mathcal{NP}$-complete (cf. Truemper [1987]):

$$\begin{array}{ll} \textit{given}: & -\text{ a binary matroid } M \text{ on } E; \\ & -\text{ a length function } l : E \to \mathbb{R}_+; \\ & -\text{ an element } f \in E; \\ \textit{find}: & -\text{ a circuit } C \text{ of } M \text{ so that } f \in C \text{ and } l(C) \text{ is minimum.} \end{array} \tag{9.2}$$

We say that a solution to problem (9.2) is a *shortest circuit* (with respect to l) of M containing f. Johnson's results immediately implies the $\mathcal{NP}$-completeness of the problem of finding an shortest circuit basis in a binary matroid M. Indeed, let $\mathcal{B}$ be an shortest circuit basis of M and C_f a shortest circuit of M containing f. Thus,

$$C_f = C_1 \,\Delta\, \cdots \,\Delta\, C_k, \quad C_i \in \mathcal{B}\ (i = 1, \ldots, k),$$

and $f \in C_j$ for some $j \in \{1, \ldots, k\}$ implying that $l(C_j) \leq l(C_f)$ (otherwise $(\mathcal{B} \setminus \{C_j\}) \cup \{C_f\}$ is a circuit basis shorter than $\mathcal{B}$). Hence, for each $f \in E$ an shortest circuit basis of M contains a shortest circuit containing f. Therefore, by simply inspecting the circuits in an shortest circuit basis one can find a shortest circuit containing a given element $f \in E$.

This chapter is organized as follows. Section 9.2 contains known facts about binary matroids, matroid decomposition and shortest circuit bases, (We make use of these facts in Section 9.3.) Next, in Section 9.3 we show that the shortest circuit basis problem is polynomially solvable for binary matroids with no S_8-minor. In Section 9.4 we discuss the polynomially equivalence between the shortest circuit basis problem, the shortest odd circuit problem and the f-circuit problem. (We say that two problems are *polynomially equivalent* whenever there is a polynomial transformation from one to the other and vice-versa.) between two problems we mean that one can be solved in polynomial time if and only if Finally, in Section 9.5, using decomposition techniques and a claim in Section 9.3 we observe that a shortest circuit basis in a graph with no $K_{3,3}$-minor can be found in $O(m + n^2 \log n)$ time.

9.2. Preliminaries on binary matroids and an optimality criterion

We present here most of the results and notation that we shall use in the sequel of this chapter. For an introduction to matroid theory we refer to Oxley [1992], Truemper [1992] and Welsh [1976].

Let M be a binary matroid on E. The dimension of $\mathcal{C}(M)$ will be denoted

by $\nu(M)$ or simply ν. If X and Y are disjoint subsets of E, the binary matroid $M/X \setminus Y$ is said to be the matroid obtained from M by *contracting* X and *deleting* Y. A binary matroid arising in this way is called a *minor* of M, and it is a *proper minor* if $X \cup Y \neq \emptyset$. Two matroids M_1 and M_2 on E_1 and E_2, respectively are said to be *isomorphic* if there exists a bijection $\phi : E_1 \to E_2$ so that C is a circuit of M_1 if and only if $\phi(C)$ is a circuit of M_2. Finally, a matroid N' is called an N-minor of a matroid M if N' is isomorphic to a minor N of M.

$AG(3,2)$, S_8, F_7^*, F_7 and R_{10} are well-known binary matroids (see, for example, Oxley [1992]). For us what really matters is that $AG(3,2)$ and S_8 are 8-element matroids, F_7 and F_7^* are 7-element matroids and R_{10} is a 10-element matroid. Hence, an shortest circuit basis either one of these matroids can be found by complete enumeration in constant time.

Decomposition of matroids

Let M_1 and M_2 be binary matroids on E_1 and E_2, respectively. The binary matroid M on $E_1 \Delta E_2$ whose circuits are the minimal nonempty subsets in $\{C_1 \Delta C_2 \mid C_i \text{ is a circuit of } M_i \ (i = 1,2)\}$ is said to be the:

(*i*) *1-sum* of M_1 and M_2 if $E_1 \cap E_2 = \emptyset$;

(*ii*) *2-sum* of M_1 and M_2 if $|E_1|, |E_2| \geq 2$, and $E_1 \cap E_2 = \{e\}$, and e is neither a loop of M_1 nor of M_2 (a *loop* is a circuit with just one element);

(*iii*) *3-sum* of M_1 and M_2 if $E_1 \cap E_2 = \{e_1, e_2, e_3\}$ and $\{e_1, e_2, e_3\}$ is a circuit of both M_1 and M_2.

The k-sum of M_1 and M_2 is denoted by $M_1 \oplus_k M_2$ and we say that M_1 and M_2 are *components* of a *k-sum decomposition* of M. (The 3-sum here corresponds to the Δ-sum in Truemper [1992].) A matroid is *k-connected* ($k = 2,3$) if it can not be written as a k'-sum decomposition of its proper minors with $1 \leq k' < k$.

It is helpful to visualize the k-sum decomposition in terms of graphic matroids. Suppose $M = M_1 \oplus_k M_2$ is the graphic matroid of a graph G. Then M_i is a graphic matroid of a graph G_i ($i = 1,2$) and the graph G can be constructed from G_1 and G_2 as suggested by Figure 9.1.

Fundamental ingredients in the algorithm we present are the following famous decomposition theorems of regular matroids and matroids with no S_8-minor due to Seymour [1980,1985] (see also Truemper [1992: Ch. 11, 12 and 13] and Oxley [1992: Ch. 11 and 13]). A matroid is *regular* if it contains neither an F_7- nor an F_7^*-minor.

Theorem 9.1 (Seymour [1980,1985]) *Every 3-connected binary matroid with no S_8-minor is:*

either (*i*) *isomorphic to F_7, F_7^* or $AG(2,3)$;*

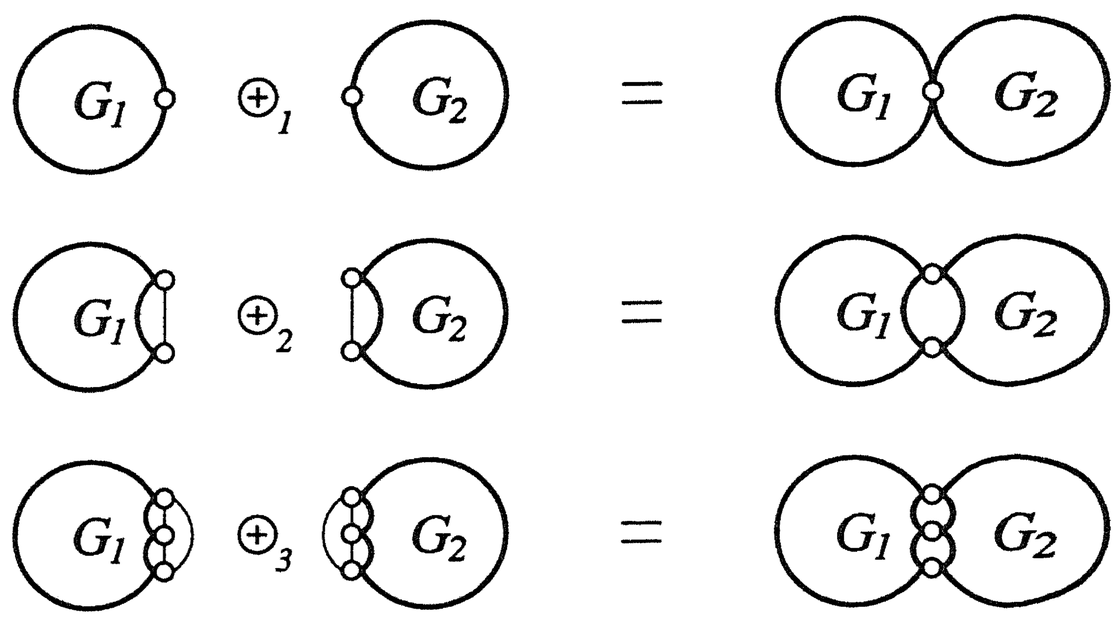


Figure 9.1: Drawings illustrating the k-sum operation in graphs ($k = 1, 2, 3$).

or (*ii*) *is regular.*

∎

Theorem 9.2 (Seymour [1980]) *Every regular matroid M is:*
(*i*) *isomorphic to R_{10};*
or (*ii*) *is graphic or cographic;*
or (*iii*) *can be decomposed into proper minors of M by a 1-, 2- or 3-sum decomposition.*

∎

Polynomial-time algorithms for determining whether or not a given binary matroid is graphic or cographic can be found in, for example, Bixby & Cunningham [1980], Bixby & Wagner [1988] and Tutte [1960]. Cunningham [1973] and Cunningham & Edmonds [1980] observed that a decomposition as stated in Seymour's theorems can be found in polynomial time with the "matroid intersection algorithm" (see also Truemper [1992: Ch. 5]; for a more efficient method see Truemper [1992: Ch. 6]).

The f-circuit problem on regular matroids

A *signing* of a $\{0,1\}$-matrix is the matrix resulting by replacing some of the 1's by -1's. As it is well-known a binary matroid M is regular if any of its representation matrices can be signed to become *totally unimodular* (when considered to be a matrix over $\mathbb{R}$), i.e. every square submatrix has determinant 0, $+1$ or -1. A "totally unimodular" signing of a representation matrix of a regular matroid can be found in polynomial time (see, for instance, Schrijver [1986: Ch. 21]). Let us call the resulting matrix a *totally unimodular representation* of M.

Given a totally unimodular representation matrix A of a regular matroid M the shortest f-circuit problem can be solved in polynomial time as follows

(cf. Truemper [1987]). Find an optimal solution (w^+, w^-) for the linear program:

$$\min\{l^T x^+ + l^T x^- \mid Ax^+ - Ax^- = 0,\ x_f^+ = 1, x_f^- = 0,\ x^+, x^- \geq 0\},$$

by a (strongly) polynomial-time algorithm (cf. Tardos [1986]). By Hoffman & Kruskal's theorem we may assume that (w^+, w^-) is integral (see Schrijver [1986: Thm. 16.2 and Thm. 19.2a]). Now, $C_f := \{e \in E \mid w_e^+ + w_e^- = 1\}$ is a shortest f-circuit of M (relative to the length function l).

An optimality criterion

In the sequel we shall make use of the following characterization of an shortest circuit basis.

Proposition 9.1 *Let $\mathcal{B}$ be a circuit basis of M and $l : E \to \mathbb{R}_+$. Then $\mathcal{B}$ is an shortest circuit basis of M if and only if*

for each circuit C of M, $C = C_1 \,\Delta\, \cdots \,\Delta\, C_k$, $C_i \in \mathcal{B}$ $(i = 1, \ldots, k)$ implies that $l(C_i) \leq l(C)$ $(i = 1, \ldots, k)$.

Proof. The necessity being clear we prove sufficiency. Let $\mathcal{B}$ be a circuit basis of M fulfilling the conditions of the proposition. Suppose by contradiction that $\mathcal{B}$ is not an shortest circuit basis and let $\mathcal{B}'$ be an shortest circuit basis of M so that $|\mathcal{B} \cap \mathcal{B}'|$ is maximum. Let $C' \in \mathcal{B}' \setminus \mathcal{B}$. Since $\mathcal{B}$ is a basis we have that

$$C' = C_1 \,\Delta\, \cdots \,\Delta\, C_k, \quad C_i \in \mathcal{B}\ (i = 1, \ldots, k).$$

There exists an index $j \in \{1, \ldots, k\}$ so that $C_j \notin \mathcal{B}'$. As $l(C') \geq l(C_j)$, $\mathcal{B}'' := (\mathcal{B}' \setminus \{C'\}) \cup \{C_j\}$ is an shortest circuit basis of M so that $|\mathcal{B} \cap \mathcal{B}''| > |\mathcal{B} \cap \mathcal{B}'|$, a contradiction. ∎

9.3. Shortest bases of circuits in matroids with no S_8-minor

In the present section we prove the following theorem.

Theorem 9.3 *The shortest circuit basis problem is polynomially solvable for binary matroids with no S_8-minor.*

Proof. Let M be a binary matroid on E and $l : E \to \mathbb{R}_+$. In polynomial time it can be tested whether or not M contains a S_8-minor by proceeding as follows. Firstly, we decompose M into its 3-connected components (see Truemper [1992: Ch. 8]). Secondly, by Theorem 9.1, it suffices to check whether each of these components is either isomorphic to F_7, F_7^* or $AG(3,2)$ or is regular (see Truemper [1992: Ch. 11]). This decomposition and checking can be carried out in polynomial time.

We may assume that $l(e) > 0$ for all $e \in E$, e not a loop of M, otherwise we can translate an shortest circuit basis of the matroid $M/\{e\}$ into an shortest circuit basis of M.

Below we either:

(*i*) find an shortest circuit basis of M; or
(*ii*) reduce M (via a k-sum decomposition) to proper minors M_1 and M_2 and show how shortest circuit bases of M_1 and M_2 can be combined in order to give us an shortest circuit basis of M.

By Theorem 9.1 and 9.2 we have to consider three possibilities. The applicable case is selected using the polynomial-time algorithms cited in Section 9.2.

Case 1. M is isomorphic to either F_7, F_7^*, $AG(3,2)$ or R_{10}. This case is handled by enumeration.

Case 2. M is graphic (cographic, respectively).

Find a graph $G = (V, E)$ for the given matroid M so that M is represented by A_G (A_G^*, respectively). This can be done (for instance) by the "almost" linear-time algorithm of Bixby & Wagner [1988]. Then any shortest circuit (cut, respectively) basis of G (relative to l) is an shortest circuit basis M. A shortest circuit (cut, respectively) basis of G can be found in polynomial time as described in Section 7.2 (Section 8.2, respectively).

Case 3. M can be decomposed into proper minors.

By Theorem 9.1 and 9.2 we need to consider three possibilities.

Case 3.1. $M_1 \oplus_1 M_2$ is a 1-sum decomposition of M. Find an shortest circuit basis $\mathcal{B}_i$ of M_i ($i = 1, 2$). $\mathcal{B}_1 \cup \mathcal{B}_2$ is an shortest circuit basis of M.

Case 3.2. $M_1 \oplus_2 M_2$ is a 2-sum decomposition of M.

Let E_i be the ground set of M_i ($i = 1, 2$) and $\{f\} := E_1 \cap E_2$. Without loss of generality we may assume that M_1 is 3-connected. Define $w_1 : E_1 \to \mathbb{R}_+$ so that $w_1(f) := 0$ and $w_1(e) := l(e)$ for all $e \in E_1 \setminus \{f\}$. Find a shortest circuit $C_{1,f}$ of M_1 containing f (relative to w_1). By Theorem 9.1 and a result in Truemper [1987] this can be done in polynomial time. Define the length function $l_2 : E_2 \to \mathbb{R}_+$ as follows:

$$l_2(e) := \begin{cases} w_1(C_{1,f}) & e = f; \\ l(e) & e \in E_2 \setminus \{f\}. \end{cases}$$

Solve the shortest circuit basis problem for M_2 with length function l_2, getting an shortest circuit basis $\mathcal{B}_2$. Now, let $C_{2,f} \in \mathcal{B}_2$ be a shortest circuit of M_2 containing f (relative to l_2). Define the length function $l_1 : E_1 \to \mathbb{R}_+$ as follows:

$$l_1(e) := \begin{cases} l_2(C_{2,f}) - l_2(f) & e = f; \\ l(e) & e \in E_2 \setminus \{f\}. \end{cases}$$

Solve the shortest circuit basis problem for M_1 with length function l_1, getting an shortest basis $\mathcal{B}_1$. We may assume that $C_{1,f}$ is a circuit in $\mathcal{B}_1$. Let $\mathcal{B}$ the

set of all $C \subseteq E_1 \Delta E_2$ satisfying (i) or (ii) below:

(i) $C \in \mathcal{B}_i$ for some $i \in \{1,2\}$;
(ii) $C = C_i \Delta C_{3-i,f}$, where $C_i \in \mathcal{B}_i$ for some $i \in \{1,2\}$.

The following claim completes the proof of the present case.

Claim 9.1 *$\mathcal{B}$ is an shortest circuit basis of M.*

Proof. It is easy to see that each $C \in \mathcal{B}$ is a circuit of M. Observe also that $|\mathcal{B}| = |\mathcal{B}_1| + |\mathcal{B}_2| - 1 = \nu(M)$, the dimension of the circuit space of M. (The '-1' comes from the assumption that $C_{f,i}$ is a circuit in $\mathcal{B}_i$ $(i = 1,2)$.) We show that each circuit C of M can be written as symmetric difference of circuits in $\mathcal{B}$ whose length is less than or equal to the length of C, by Proposition 9.1 and the previous observation this proves the claim. For every circuit $C \in \mathcal{B}_i$ we shall denote by C' the corresponding circuit in $\mathcal{B}$.

Let C be a circuit of M. We have two possibilities to consider.

Case (i). C is a circuit of M_i for some $i \in \{1,2\}$.

Assume $i = 1$. Since $\mathcal{B}_1$ is an shortest circuit basis of M_1 we have that

$$C = C_1 \Delta \cdots \Delta C_k,$$

where $C_1, \ldots, C_k$ are circuits in $\mathcal{B}_1$ satisfying $l_1(C_i) \leq l_1(C) = l(C)$ $(i = 1, \ldots, k)$. Thus,

$$C = C'_1 \Delta \cdots \Delta C'_k,$$

where $C'_1, \ldots, C'_k$ are circuits in $\mathcal{B}$ satisfying $l(C'_i) = l_1(C_i) \leq l(C)$ $(i = 1, \ldots, k)$.

Case (ii). $C \cap E_i \neq \emptyset$ $(i = 1,2)$.

Let $C_i := (\{C \cap E_i\}) \cup \{f\}$ $(i = 1,2)$. (Observe that $l_i(C_i) \leq l(C)$ $(i = 1,2)$.) Since $\mathcal{B}_i$ is an shortest circuit basis of M_i we have that for $i = 1,2$ the following holds:

$$C_i = C_{i,1} \Delta \cdots \Delta C_{i,k_i},$$

where $C_{i,1}, \ldots, C_{i,k_i}$ are circuits in $\mathcal{B}_i$ satisfying $l_i(C_{i,j}) \leq l_i(C_i) \leq l(C)$ $(j = 1, \ldots, k_i;\ i = 1,2)$. Hence,

$$C = C'_{1,1} \Delta \cdots \Delta C'_{1,k_1} \Delta C'_{2,1} \Delta \cdots \Delta C'_{2,k_2} \Delta \{C_{1,f} \Delta C_{2,f}\},$$

where $C'_{1,1}, \ldots, C'_{1,k_1}, C'_{2,1}, \ldots, C'_{2,k_2}, \{C_{1,f} \Delta C_{2,f}\}$ are all circuits in $\mathcal{B}$ as least as short as C. ∎

Case 3.3. $M_1 \oplus_3 M_2$ is a 3-sum decomposition of M.

We proceed similarly to Case 3.2. The only difference is that in the present case after combining shortest circuit bases of the proper minors M_1 and M_2 into a set $\mathcal{F}$ of *cycles* of M we still have to apply the greedy algorithm to $\mathcal{F}$ in order to obtain an shortest circuit basis of M. We may assume that M is 3-connected, otherwise we could apply Case 3.2. So, M_1 and M_2 are also 3-connected.

Let E_i be the ground set of M_i $(i = 1, 2)$ and $F := E_1 \cap E_2$. Define for $i = 1, 2$ and $f \in F$ the length function $w_{i,f} : E_i \to \mathbb{R}_+$ as follows:

$$w_{i,f}(e) := \begin{cases} 0 & e = f \\ L & e \in F \setminus \{f\} \\ l(e) & e \in E_i \setminus F, \end{cases}$$

where L is an integer so that for each $f \in F$ the shortest circuit $C_{i,f}$ of M_i containing f relative to the length $w_{i,f}$ satisfies $C_{i,f} \cap F = \{f\}$ (note that there exists a circuit satisfying this condition, otherwise M would have a 2-sum decomposition). By Theorem 9.1 and a result in Truemper [1987] such a circuit $C_{i,f}$ can be found in polynomial time $(i = 1, 2;\ f \in F)$. Now, define for $i = 1, 2$ the length function $l_i : E_i \to \mathbb{R}_+$ as follows:

$$l_i(e) := \begin{cases} w_{3-i,f}(C_{3-i,f}) & e = f \in F; \\ l(e) & e \in E_i \setminus F. \end{cases}$$

For $i = 1$ and $i = 2$, find a shortest circuit basis $\mathcal{B}_i$ with respect to l_i in M_i. We define the set $\mathcal{F}$ of all $C \subseteq E_1 \,\Delta\, E_2$ so that:

(*i*) $C \cap F = \emptyset$ and $C \in \mathcal{B}_i$ for some $i \in \{1, 2\}$;
(*ii*) $C = C_i \,\Delta\, (\Delta_{f \in C_i \cap F}\, C_{3-i,f})$, where C_i is a circuit of M_i in $\mathcal{B}_i$ for some $i \in \{1, 2\}$.

Now we prove the following.

Claim 9.2 *$\mathcal{F}$ contains an shortest circuit basis of M.*

Note that $|\mathcal{F}| \leq |\mathcal{B}_1| + |\mathcal{B}_2| \leq \nu(M) + 4$. Thus, Claim 9.2 implies that an shortest circuit basis of M can be found in polynomial time by applying the greedy algorithm to $\mathcal{F}$. It is possible that some $C \in \mathcal{F}$ is not a circuit of M (by definition C is certainly a cycle of M). However, every cycle in an shortest basis is automatically a circuit of M as $l(e) > 0$ for all $e \in E$. So, the greedy algorithm will give as output an basis formed by circuits of M. For every $C \in \mathcal{B}_i$ we shall denote by C' the corresponding cycle in $\mathcal{F}$.

Proof of Claim 9.2. By Proposition 9.1 it suffices to show that every circuit C of M can be written as symmetric difference of cycles in $\mathcal{F}$ whose lengths are less than or equal to the length of C.

Let C be a circuit of M. We have two possibilities to consider.

Case (i). C is a circuit of M_i for some $i \in \{1,2\}$.

This case is analogous to Case (i) of Claim 9.1.

Case (ii). $C \cap E_i \neq \emptyset$ $(i = 1,2)$.

In this case there exists a $f \in F$ so that $C_i := (C \cap E_i) \cup \{f\}$ is a circuit of M_i $(i = 1,2)$. We may assume that $l(C_{2,f} \setminus \{f\}) \leq l(C_{1,f} \setminus \{f\})$. Since $\mathcal{B}_1$ is an shortest basis of M_1 then

$$C_1 \;=\; C_{1,1} \,\Delta\, \cdots \,\Delta\, C_{1,k_1},$$

where $C_{1,1}, \ldots, C_{1,k_1}$ are circuits of M_1 in $\mathcal{B}_1$ satisfying $l_1(C_{1,i}) \leq l_1(C_1) \leq l(C)$ $(i = 1, \ldots, k_1)$. Thus,

$$C_1 \,\Delta\, C_{2,f} \;=\; C'_{1,1} \,\Delta\, \cdots \,\Delta\, C'_{1,k_1},$$

where $C'_{1,1}, \ldots, C'_{1,k_1}$ are cycles of M in $\mathcal{F}$ so that $l(C_{1,i}) \leq l_1(C_{1,i}) \leq l(C)$ $(i = 1, \ldots, k_1)$. Now, $C_2 \,\Delta\, C_{2,f}$ is a cycle in M_2 and therefore

$$C_2 \,\Delta\, C_{2,f} \;=\; C_{2,1} \,\Delta\, \cdots \,\Delta\, C_{2,k_2},$$

where $C_{2,1}, \ldots, C_{2,k_2}$ are circuits of M_2 in $\mathcal{B}_2$ satisfying $l_2(C_{2,i}) \leq l_2(C_2 \,\Delta\, C_{2,f})$ $\leq l(C)$ $(i = 1, \ldots, k_2)$. Hence,

$$C_2 \,\Delta\, C_{2,f} \;=\; C'_{2,1} \,\Delta\, \cdots \,\Delta\, C'_{2,k_2},$$

where $C'_{2,1}, \ldots, C'_{2,k_2}$ are cycles of M in $\mathcal{F}$ with $l(C'_{2,i}) \leq l_2(C_{2,i}) \leq l(C)$ $(i = 1, \ldots, k_2)$. Combining the above equalities we obtain that

$$\begin{aligned} C &= C_1 \,\Delta\, C_2 \\ &= C'_{1,1} \,\Delta\, \cdots \,\Delta\, C'_{1,k_1} \,\Delta\, C'_{2,1} \,\Delta\, \cdots \,\Delta\, C'_{2,k_2}, \end{aligned}$$

where $C'_{1,1}, \ldots, C'_{1,k_1}, C'_{2,1}, \ldots, C'_{2,k_2}$ are all cycles of M in $\mathcal{F}$ at least as short as C. ∎

This completes the analysis of Case 3.3 and the proof of Theorem 9.3. ∎

9.4. Shortest bases of circuits, shortest odd circuits and shortest f-circuits

Let us consider the natural extension of the shortest odd circuit problem to binary matroids (*i.e. given:* a binary matroid M on E; a nonnegative length function l on E; and a subset Σ of E; *find:* a circuit C of M so that $|C \cap \Sigma|$ is odd and $l(C)$ is as small as possible.). One can use a polynomial-time algorithm for the odd circuit problem in an algorithm analogous to the shortest circuit basis algorithm, presented in Section 7.2, in order to obtain a polynomial-time algorithm for the shortest circuit basis problem. Conversely, given a

polynomial-time algorithm for the shortest circuit basis problem one can find a shortest odd circuit by first computing an shortest circuit basis $\mathcal{B}$ and then chose the shortest circuit $C \in \mathcal{B}$ so that $|C \cap \Sigma|$ is odd. Hence, we have established the following.

Theorem 9.4 *For binary matroids the following problems are polynomially equivalent:*

(*i*) *the shortest circuit basis problem; and*
(*ii*) *the shortest odd circuit problem.*

∎

Analogous results to the ones obtained in section 9.2.1 establishing the relationship between circuit bases and odd circuits in signed graphs hold here as well.

Given an shortest circuit basis of a binary matroid M on E one can in polynomial time find a shortest circuit of M containing a given element $f \in E$, as we have already observed. On the other hand, from the proof of Theorem 9.3 it follows that if we are given a polynomial-time algorithm for the shortest f-circuit problem then one can, by using decomposition techniques and the greedy algorithm, solve the shortest circuit basis problem in polynomial time for binary matroid with no S_8-minor. Thus, we have established the following corollary.

Corollary 9.1 *For binary matroids with no S_8-minor the following problems are polynomially equivalent:*

(*i*) *the shortest circuit basis problem (9.1);*
(*ii*) *the shortest odd circuit problem; and*
(*iii*) *the shortest f-circuit problem (9.2).*

∎

9.5. Further notes: shortest bases of circuits in graphs with no $K_{3,3}$-minor

We present here a divide-and-conquer algorithm to find a shortest circuit basis of a graph G with no $K_{3,3}$-minor in $O(m + n^2 \log n)$ time, where n and m are the number of vertices and edges of G, respectively. In order to do so we shall combine:

(*i*) the algorithmic proof of Case 3.2 of Theorem 9.3;
(*ii*) a linear-time algorithm to compute a "t-decomposition" of a graph; and

(*iii*) a decomposition theorem for graphs with no $K_{3,3}$-minor due to Hall [1943].

Actually, this very elementary divide-and-conquer strategy reduces the time complexity of finding a shortest circuit basis for a graph G to the time complexity of finding the shortest circuit basis for its 3-connected components.

We first describe the known results on (*ii*) and (*iii*) needed by the algorithm.

t-decomposition

Bixby & Wagner [1988] introduced the notion of a t-decomposition; which is a small variation on a theory developed by Tutte [1968: Ch. 11]. We outline here this notion.

A *bond* is a connected loopless graph on two vertices. Let $G = (V, E)$ be a connected graph and let $\{E_1, E_2\}$ be a partition of E. Then, $\{E_1, E_2\}$ is a *k-separation* of G ($k > 0$) if:

$$|E_1| \geq k, \quad |E_2| \geq k \quad \text{and} \quad |V(G[E_1]) \cap V(G[E_2])| = k.$$

The graph G is said to be n-connected ($n \geq 1$) if it has no k-separation for $k < n$.

Roughly speaking, a *t-decomposition* is a rooted arborescence whose vertices are graphs; each one been a bond, a circuit or a 3-connected graph with at least 3 edges. Two vertices, that are graphs, are adjacent if they have an edge in common, called "marker" edge. In addition, no two bonds or two circuits have a marker edge in common. Figure 9.2 illustrates a t-decomposition. If

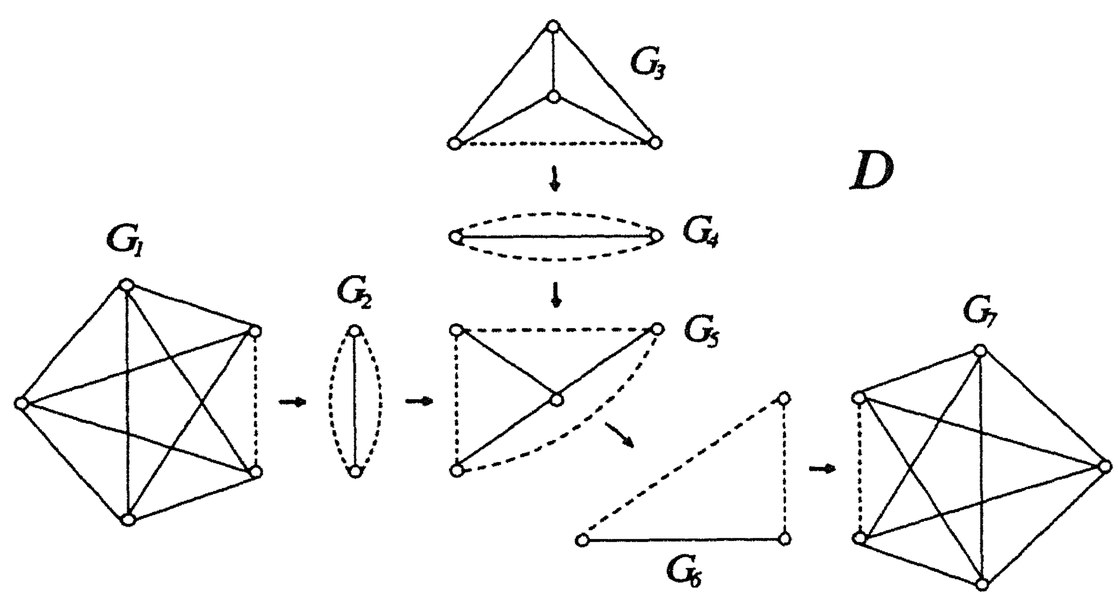


Figure 9.2: A t-decomposition. Arcs between graphs indicate the pairings of the marker edges (arcs of the arborescence). The marker edges are indicated by dashed lines.

(H, K) is an arc of a t-decomposition so that $E(H) \cap E(K) = \{e\}$, then e is

the *child marker* of K and the *parent marker* of H; this latter is denoted by $pm(H)$.

Associated with a t-decomposition D there is a *merged graph* $m(D)$ obtained by identifying the common marker edges and then erasing them. The merged graph for the graph in Figure 9.2 is shown in Figure 9.3. Given a

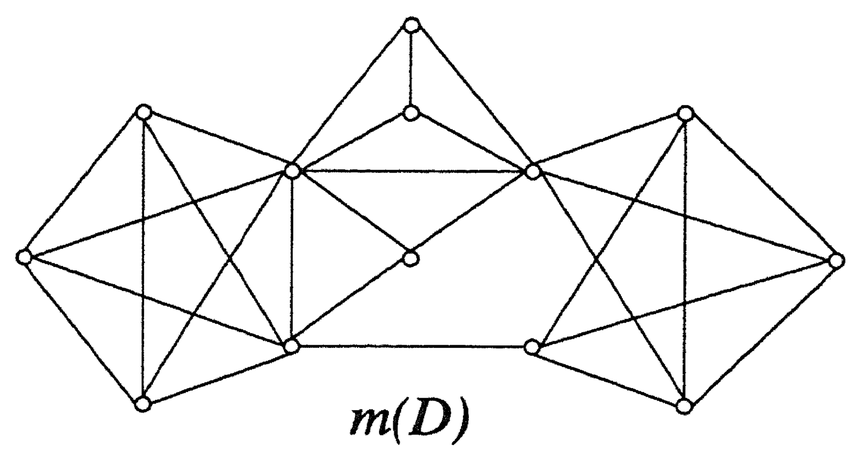


Figure 9.3: The merged graph for the t-decomposition in Figure 9.2.

2-connected graph G a t-decomposition D so that $m(D) = G$ can be found in linear time (cf. Hopcroft & Tarjan [1973]).

We also associate with each marker edge f of a t-decomposition D of G two t-decompositions $D_{f,1}$ and $D_{f,2}$ in a very natural way; namely, $D_{f,1}$ and $D_{f,2}$ are the two arborescences resulting by deleting the arc of D corresponding to f. So, $G_1 := m(D_{f,1})$ and $G_2 := m(D_{f,2})$ are minors of G so that $G = G_1 \oplus_2 G_2$ (meaning that the graphic matroid of G is the 2-sum decomposition of the graphic matroid of G_1 and G_2; see Figure 9.1).

The following can easily be proved by induction on the number of vertices of G.

Let $G = (V, E)$ be a 2-connected graph with $|V| \geq 3$. Let D be a t-decomposition of G and let $G_1, \ldots, G_t$ be the 3-connected members of D. Then

$$\sum_{i=1}^{t} |V(G_i)| \leq 3|V| - 6. \tag{9.3}$$

By induction on the number of edges of G one can also easily prove:

Let $G = (V, E)$ be a 2-connected graph with $|E| \geq 3$. Let D be a t-decomposition of G and let $G_1, \ldots, G_t$ be the members of D. Then

$$\sum_{i=1}^{t} |E(G_i)| \leq 3|E| - 6. \tag{9.4}$$

Hall's decomposition theorem

For a proof of the following theorem see Oxley [1992: Prop. 11.2.12].

Theorem 9.5 (Hall [1943]) *Let G be a 3-connected simple graph. Then G has no $K_{3,3}$-minor if and only if either G is planar or G is isomorphic to K_5.* ∎

In the sequel we also use the following remark:

Let $G = (V, E)$ be a simple graph with no $K_{3,3}$-minor. Then $|E| \leq 3|V| - 5$. (9.5)

The above remark can easily be proved using Hall's theorem by induction on the number of vertices of G.

The algorithm

Let us be given an undirected 2-connected graph $G = (V, E)$ and a nonnegative perturbed length function $l : E \to \mathbb{R}_+$ (a length function on E is said to be *perturbed* if no two subsets of E have the same length). We find a shortest circuit basis of G as follows:

Step 1. Determine a t-decomposition D of G.

Step 2. For each marker edge f let $D_{f,1}$ and $D_{f,2}$ be the t-decompositions associated with f. Compute the shortest path $P_{f,i}$ between the end-points of f in the graph $m(D_{f,i}) \setminus \{f\}$ $(i = 1, 2)$.

Step 3. For each member H of D proceed as follows. Let $f_1, \ldots, f_t$ be the marker edges of H and let $j_1, \ldots, j_t$ be indices so that $j_i \in \{1, 2\}$ and H is not a member of D_{f_i, j_i} $(i = 1, \ldots, t)$. Define the length function $l_H : E(H) \to \mathbb{R}_+$ as follows:

$$l_H(e) := \begin{cases} l(e) & e \in E; \\ l(P_{f_i,j_i}) & e = f_i \text{ for some } i \in \{1, \ldots, t\}. \end{cases}$$

Find the shortest circuit basis $\mathcal{B}_H$ of H with respect to the length function l_H.

Step 4. Let H be the root of D and let $G_1, \ldots, G_t$ be the others members of D. Let C_i be the shortest $pm(G_i)$-circuit in $\mathcal{B}_{G_i}$. The algorithm returns

$$\mathcal{B} := \mathcal{B}_H \cup (\bigcup_{i=1}^{t} (\mathcal{B}_{G_i} \setminus \{C_i\})).$$

Correctness of the algorithm

We first observe that some circuits in $\mathcal{B}$ contain marker edges. For each marker edge we use a pointer to the corresponding shortest path in G, computed in

Step 2. So, we may say that some circuits in $\mathcal{B}$ are a compact or encoded representation of circuits of G. We also remark that the algorithm's assumption that the given graph is 2-connected is no restriction since the 2-connected components of a graph can be determined in linear-time (cf. Tarjan [1972]).

The correctness of the algorithm follows directly from the proof of Case 3.2 of Theorem 9.3. This can be seen by observing the following.

Let H and K be members of D so that f is the child marker of K and the parent marker of H. Let $\mathcal{B}_K$ and $\mathcal{B}_H$ be shortest circuit basis of K and H relative to the (perturbed) length functions l_K and l_H defined in Step 3, respectively, and let C_f be *the* shortest f-circuit of H. Then $\mathcal{B} := \mathcal{B}_K \cup (\mathcal{B}_H \setminus \{C_f\})$ is precisely the set of circuits of the merge graph $m(H,K)$ of H with K $(= H \oplus_2 K)$ as defined in the proof of Case 3.2 of Theorem 9.3. So, $\mathcal{B}$ is a shortest circuit basis of $m(H,K)$ (relative to the restriction of l_H and l_K to $E(m(H,K))$).

Complexity of the algorithm

We denote by m and n the number of edges and the number of vertices of the given graph $G = (V,E)$. Step 1 takes $O(m+n)$ time (cf. Hopcroft & Tarjan [1973]). By (9.4), Step 2 takes $O(mT_{ss}(m,n))$ time (recall that $T_{ss}(m,n)$ denotes the function expressing the time complexity for solving the single source shortest path problem in a graph with m edges and n vertices). The complexity to find a shortest circuit basis in a bond or a circuit is bounded by the number of edges of the bond or circuit. Let $G_1, \ldots, G_t$ be the 3-connected members of the t-decomposition of G. Hence, Step 3 can be performed in time $O(m + \sum_{i=1}^{t} S(G_i))$, where $S(H)$ is the function expressing the time complexity for finding a shortest circuit basis for the graph H. Step 4 takes $O(\nu(H)+\nu(G_1)+\cdots\nu(G_t)) = O(m)$ by (9.4) (if we keep the length of each circuit in $\mathcal{B}_i$ then the shortest $pm(G_i)$-circuit in $\mathcal{B}_i$ can be found in $O(\nu(G_i))$ time). Therefore, the running time of the algorithm is $O(mT_{ss}(m,n) + \sum_{i=1}^{t} S(G_i))$.

Complexity of the algorithm for graphs with no $K_{3,3}$-minor

Let us now specialize the previous analysis to graphs with no $K_{3,3}$-minor. Let $G = (V,E)$ be a given simple graph (we observe latter how to deal with the nonsimple graphs) with m edges and n vertices and let l the given perturbed length function on E.

One can test in linear time whether a graph has a $K_{3,3}$-minor. Indeed, a t-decomposition D of a graph G can be determined in linear time. Now, by Hall's theorem, we have only to check whether each one of the 3-connected members of D is either planar (one can determining whether a graph is planar in linear time; cf. Hopcroft & Tarjan [1974]) or isomorphic to K_5 (up to parallel edges).

From remarks (9.4) and (9.5) it follows that $m = O(n)$. Thus, Step 1

takes $O(n)$ time and Step 2 takes $O(n^2 \log n)$ time, using (say) the implementation of Dijkstra's algorithm due to Johnson [1977]. Let $G_1, \ldots, G_t$ be the 3-connected members of the t-decomposition of G computed in Step 1. By Hall's theorem G_i is either a planar graph or isomorphic to K_5 ($i = 1, \ldots, t$). In both cases the time complexity of finding a shortest circuit basis for G_i is $O(|V(G_i)|^2 \log |V(G_i)|)$ ($i = 1, \ldots, t$) (see, e.g., Hartvigsen & Mardon [1994]). Thus, Step 3 takes

$$O(\sum_{i=1}^{t}(|V(G_i)|^2 \log |V(G_i)|) \text{ time.}$$

From remark (9.3) we have that

$$\sum_{i=1}^{t} |V(G_i)|^2 = O(n^2).$$

Hence, the time complexity of Step 3 is $O(n^2 \log n)$ time . Finally, Step 4 can be performed in time $O(n)$. Therefore, the overall running time of the algorithm for simple graphs with no $K_{3,3}$-minor is $O(n^2 \log n)$.

As observed by Hartvigsen & Mardon [1994] the shortest circuit basis problem on nonsimple (not necessarily planar) graphs can be easily reduced to the shortest circuit basis on a simple graphs by preprocessing the given graph. This preprocessing procedure is as follows. Let $G = (V, E)$ be the given graph. Initially set $E' := E$ and $\mathcal{B}' := \emptyset$. For each class $\{e_1, \ldots, e_t\}$ ($t \geq 2$) of parallel edges of G where $l(e_1) \leq \ldots \leq l(e_t)$ set $E' := E' \setminus \{e_2, \ldots, e_t\}$. For each pair of vertices u and v of V compute a shortest $u - v$-path $P_{u,v}$ in $G[E']$. For each class of parallel edges $e_1, \ldots, e_t$ ($t \geq 2$) with endpoints (say) u and v where $e_1 \in E'$ set

$$\mathcal{B}' := \mathcal{B}' \cup \{P_{u,v} \cup \{e_2\}, \ldots, P_{u,v} \cup \{e_t\}\}.$$

The algorithm returns $\mathcal{B}'$. (We note that one does not need to write out the shortest $u - v$-path $P_{u,v}$ for each edge e_i ($i = 2, \ldots, t$). A pointer from e_i to $P_{u,v}$ suffices ($i = 2, \ldots, t$). Thus, $\mathcal{B}'$ can be compactly written out in $O(m)$ time.)

Note that if $\mathcal{B}$ is a shortest circuit basis for $G[E']$ then $\mathcal{B} \cup \mathcal{B}'$ is a shortest circuit basis for G. This preprocessing procedure can easily be seen to run in time $O(m + n^2 \log n)$ in graphs with $O(n)$ parallel classes.

So, in particular, the combination of this preprocessing procedure with the above algorithm leads to a $O(m + n^2 \log n)$-time algorithm for the shortest circuit basis problem in graphs with no $K_{3,3}$-minor. We state the discussion above as the following theorem.

Theorem 9.6 *Solving the shortest circuit basis problem (7.1) in graphs with no $K_{3,3}$-minor requires $O(m + n^2 \log n)$ time.* ∎

References

Adler, I. & Cosares, S. (1991), A strongly polynomial algorithm for a special class of linear programs, *Operations Research* **39** (1991) 955–960. [56]

Aho, A.V., Hopcroft, J.E. & Ullman, J.D. (1974), *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading, MA, 1974. [12, 13, 14]

Ahuja, R.K., Magnanti, T.L. & Orlin, J.B. (1993), *Network Flows*, Prentice-Hall, Englewood Cliffs, NJ, 1993. [5, 17, 27, 45, 56, 62, 65, 126, 127, 133]

Ahuja, R.K., Magnanti, T.L., Orlin, J.B. & Reddy, M.R. (1995), Applications of network optimization, [in: Network Models (Ball, M.O., Magnanti, T.L. & Monma, C.L. eds.)] *Handbooks in Operations Research and Management Sciences* **7** (1995) 1–83. [5]

Ahuja, R.K., Mehlhorn, K., Orlin, J.B. & Tarjan, R.E. (1990), Faster algorithms for the shortest path problem, *Journal of the Association for Computing Machinery* **37** (1990) 213–223. [18]

Bellman, R. (1958), On a routing problem, *Quarterly of Applied Mathematics* **16** (1958) 87–90. [18, 42]

Bixby, R.E. & Cunningham, W.H. (1980), Converting linear programs to network problems, *Mathematics of Operations Research* **5** (1980) 321–357. [138]

Bixby, R.E. & Wagner, D.K. (1988), An almost linear-time algorithm for graph realization, *Mathematics of Operations Research* **13** (1988) 99–123. [138, 140, 145]

Bollobás, B. (1979), *Graph Theory. An Introductory Course,* Graduated Texts in Mathematics, 63, Springer-Verlag, Berlin, 1979. [9, 117]

Bondy, J.A. & Murty, U.S.R. (1976), *Graph Theory with Applications*, Macmillan, London, 1976. [9]

Bryant, P.R. (1967), Graph theory applied to electrical networks, in: *Graph Theory and Theoretical Physics* (F. Harary, ed.), Academic Press, Norwich, 1967, pp. 111–137. [117]

Charnes, A. & Raike, W.M. (1966), One-pass algorithm for some generalized network problems, *Operations Research* **14** (1966) 914–924. [58]

Chua, L.O. & Chen L.K. (1973), On optimally sparse cycle and coboundary basis for a linear graph, *IEEE transactions on circuit theory* **CT-20** (1973) 495–503. [117]

Cobham, A. (1965), The intrinsic computational difficulty of functions, in: *Logic, Methodology and Philosophy of Science* (Proceedings of the 1964 International Congress; Y. Bar-Hillel, ed.), North-Holland, Amsterdam, 1965, pp. 24–30. [14, 15]

Cohen, E. & Megiddo, N. (1994), Improved algorithms for linear inequalities with two variables per inequality, *SIAM Journal on Computing* **23** (1994) 1313–1347. [43, 56]

Cole, R. & Siegel, A. (1984), River routing every which way, but loose, in: *Proceedings of the 25th Annual Symposium on Foundations of Computer Science*, IEEE, 1984, pp. 65–73. [31]

Cook, S.A. (1971), The complexity of theorem-proving procedures, in: *Proceedings of Third ACM Symposium on Theory of Computing* (Shaker Heights, Ohio,1971), Association for Computing Machinery, New York, 1971, pp. 151–158. [15, 66]

Cook, S.A. & Reckhow, R.A. (1973), Time-bounded random access machines, *Journal of Computer and System Science* **7** (1973) 354–375. [13]

Cottle, R.W. & Veinott, A.F. (1972), Polyhedral sets having a least element, *Mathematical Programming* **3** (1972) 238–249. [55]

Coullard, C.R. & Pulleyblank, W.R. (1989), On cycle cones and polyhedra, *Linear Algebra and Its Applications* **114/115** (1989) 613–640. [104, 106]

Cunningham, W.H. (1973), *A combinatorial decomposition theory*, thesis, University of Waterloo, 1973. [138]

Cunningham, W.H. & Edmonds, J. (1980), A combinatorial decomposition theory, *Canadian Journal of Mathematics* **32** (1980) 734–765. [138]

Dahlhaus, E., Johnson, D.S., Papadimitriou, C.H., Seymour, P.D. & Yannakakis, M. (1994), The complexity of multiterminal cuts, *SIAM Journal on Computing* **23** (1994) 864–894. [121]

Dauscha, W., Modrow, H.D. & Neumann, A. (1985), On cyclic types for constructing cyclic schedules, *Zeitschrift für Operations Research* **29** (1985) 1–30. [86]

Deo, N. & Pang, C. (1984), Shortest-path algorithms: taxonomy and annotation, *Networks* **14** (1984) 275–323. [17]

Deo, N., Prabhu, G.M. & Krishnamoorhty, M.S. (1982), Algorithms for generating fundamental cycles in a graph, *ACM Transactions on Mathematical Software* **8** (1982) 26–42. [92, 93, 124]

Dial, R. (1969), Algorithm 360: Shortest path forest with topological ordering, *Communications of the Association for Computing Machinery* **12** (1969) 632-633. [18]

Dijkstra, E.W. (1959), A note on two problems in connexion with graphs, *Numerische Mathematik* **1** (1959) 269–271. [18]

Ding, G. (1994), Bounding the number of circuits of a graph, preprint 1994. [122]

Dreyfus, S.E. (1968), An appraisal of some shortest-path algorithms, *Operations Research* **17** (1969) 395–412. [17]

Edmonds, J. (1965), Paths, trees, and flowers, *Canadian Journal of Mathematics* **17** (1965) 449–467. [15]

Edmonds, J. & Karp, R.M. (1972), Theoretical improvements in algorithmic efficiency for network flow problems, *Journal of the Association for Computing Machinery* **19** (1972) 248–264. [18]

Edmonds, J. & Giles, R. (1977), A min-max relation for submodular functions on graphs, [in: *Studies in Integer Programming* (P.L. Hammer, et al., eds.),] Annals of Discrete Mathematics **1** (1977) 185–204. [104, 130]

Even, S., Itai, A. & Shamir, A. (1976), On the complexity of timetable and multicommodity flow problems, *SIAM Journal on Computing* **5** (1976) 691–703. [23, 40, 47]

Floyd, W. (1962), Algorithm 92: Shortest path, *Communications of the Association for Computing Machinery* **5** (1962) 345. [19]

Ford, L.R., Jr (1956), *Network Flow Theory*, Paper P-923, The RAND Corporation, Santa Monica, California, 1956. [18, 42]

Fortune, S., Hopcroft, J. & Wyllie, J. (1980), The direct subgraph homeomorphism problem, *Theoretical Computer Science* **10** (1980) 111-121. [5, 23, 24]

Frederickson, G.N. (1987), Fast algorithms for shortest paths in planar graphs, with applications, *SIAM Journal on Computing* **16**, (1987) 1005–1022. [19, 134]

Fredman, M.L. & Tarjan, R.E. (1987), Fibonacci heaps and their uses in improved network optimization algorithms, *Journal of the Association for Computing and Machinery* **34** (1987) 596–615. [15, 18, 19, 58, 59]

Gabow, H.N. (1985), Scaling algorithms for networks problems, *Journal of Computer and System Sciences* **31** (1985) 148–168. [18]

Gabow, H.N. & Tarjan, R.E. (1989), Faster scaling algorithm for network problems, *SIAM Journal on Computing* **18** (1989) 1013–1036. [18]

Garey, M.R. & Johnson, D.S. (1979), *Computers and Intractability: a Guide to the Theory of $\mathcal{NP}$-completeness*, Freeman, San Francisco, 1979. [12, 14, 15, 16, 55, 69]

Goldberg, A.V. (1993), Scaling algorithms for the shortest paths problem, in: *Proceedings of the Fourth ACM-SIAM Symposium on Discrete Algorithms*, Austin, 1993, pp. 222–231. [18, 37, 42, 47]

Goldschmidt, O. & Hochbaum, D.S. (1994), A polynomial algorithm for the k-cut problem for fixed k, *Mathematics of Operations Research* **19** (1994) 24–37. [120]

Gomory, R. & Hu, T.C. (1961), Multi-terminal network flows, *Journal of the SIAM* **9** (1961) 551–570. [123, 127, 132, 133]

Grötschel, M., Lovász, L. & Schrijver, A. (1981), The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981) 169–197. [*corrigendum*: 4 (1984) 291–295]. [102]

Grötschel, M., Lovász, L. & Schrijver, A. (1988), *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Würzburg, 1988. [16, 95]

Grötschel, M., Martin, A. & Weismantel, R. (1993), Optimum path packing on wheels: the noncrossing case, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Preprint SC 93-26, 1993. [62]

Grötschel, M. & Pulleyblank, W.R. (1981), Weakly bipartite graphs and the max-cut problem, *Operations Research Letters* **1** (1981) 23–27. [95]

Gusfield, D. (1990), Very simple methods for all pairs network flow analysis, *SIAM Journal on Computing* **19** (1990) 143–155. [123, 130, 133]

Hall, D.W. (1943), A note on primitive skew curves, *Bulletin of the American Mathematical Society* **49** (1943) 935–937. [145, 147]

Hartvigsen, D. (1993), Minimum path bases, *Journal of Algorithms* **15** (1993) 125–142. [117, 118]

Hartvigsen, D. & Mardon, R. (1991), Cycle bases from ordering and covering, *Discrete Mathematics* **94** (1991) 81–94. [121]

Hartvigsen, D. & Mardon, R. (1993), When do short cycles generate the cycle space?, *Journal of Combinatorial Theory* (B) **57** (1993) 88–99. [121]

Hartvigsen, D. & Mardon, R. (1994), The all-pairs min cut problem and the minimum cycle basis problem on planar graphs, *SIAM Journal on Discrete Mathematics* **7** (1994) 403–418. [129, 134, 149]

Hartvigsen, D. & Zemel, E. (1989), Is every cycle basis fundamental?, *Journal of Graph Theory* **13** (1989) 117–137. [121]

Hassin, R. & Johnson, D.B. (1985), An $O(n \log^2 n)$ algorithm for maximum flow in undirected planar networks, *SIAM Journal on Computing* **14** (1985) 612-624. [134]

Hochbaum, D.S. & Naor, J.S. (1994), Simple and fast algorithms for linear and integer programs with two variables per inequality, *SIAM Journal on Computing* **23** (1994) 1179–1192. [43, 56]

Hopcroft, J.E. & Tarjan, R.E. (1973), Dividing a graph into triconnected components, *SIAM Journal on Computing* **2** (1973) 135–158. [146, 148]

Hopcroft, J.E. & Tarjan, R.E. (1974), Efficient planarity testing, *Journal of the Association for Computing and Machinery* **21** (1974) 549–568. [148]

Horton, J.D. (1987), A polynomial-time algorithm to find the shortest cycle basis of a graph, *SIAM Journal on Computing* **16** (1987) 358–366. [7, 91, 92, 93, 100, 101, 134]

Hu, T.C. (1982), *Combinatorial Algorithms*, Addison-Wesley, Reading, MA, 1982. [123, 127]

Hubicka, E. & Sysło, M.M. (1976), Minimal bases of cycles of a graph, in: *Recent Advances in Graph Theory* (Proceedings of the symposium held in Prague, June 1977; M. Fiedler, ed.), Academia Praha, Prague, 1976, pp. 283–293. [92, 93, 121]

Hurkens, C.A.J. (1989), On the existence of an integral potential in a weighted bidirected graph, *Linear Algebra and Its Applications* **114/115** (1989) 541–553. [40, 47, 49]

Itai, I., Perl, Y. & Shiloach, Y. (1982), The complexity of finding maximum disjoint paths with length constraints, *Networks* **12** (1982) 277–286. [65]

Itai, I. & Shiloach, Y. (1979), Maximum flow in planar networks, *SIAM Journal on Computing* **8** (1979) 135–150. [134]

Johnson, D.B. (1977), Efficient algorithms for shortest paths in sparse networks, *Journal of the Association for Computing Machinery* **24** (1977) 1–13. [18, 134, 149]

Karp, R.M. (1972), Reducibility among combinatorial problems, in: *Complexity of Computer Computations* (R.E. Miller & J.W. Thatcher, eds.), Plenum Press, New York, 1972, pp. 85–103. [15, 119]

Karzanov, A.V. (1974), Determining the maximal flow in a network by the method of preflows, *Soviet Mathematics Doklady* **15** (1974) 434–437. [126, 133]

Kaveh, A. (1989), On optimal cycle bases of graphs for mesh analysis of networks, *Networks* **19** (1989) 273–279. [93]

Khachiyan, L.G. (1979), A polynomial algorithm in linear programming (in Russian), *Doklady Akademii Nauk SSSR* **244** (1979) 1093-1096 [English translation: *Soviet Mathematics Doklady* **20** (1979) 191–194]. [16]

Kirchhoff, G. (1847), Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung Galvanische Ströme geführt wird. *Poggendorff Annalen der Physik und Chemie* **72** (1847) 497–508 [English translation: *Trans. Inst. Radio Engrs.* **CT-5** (1958) 4–7]. [115]

Klein, P., Rao, S., Rauch, M., Subramanian, S. (1994), Faster shortest-path algorithms for planar graphs, in: *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, Québec, 1994, pp. 27–37. [19]

Knuth, D.E. (1968), *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, Addison-Wiley, Reading, MA, 1968. [12, 13]

Kolasinska, E. (1980), On a minimum cycle basis of a graph, *Zastosowania Matematyki* **16** (1980) 631–639. [92]

Lawler, E. (1976), *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, New York, 1976. [17, 26, 48]

Leiserson, C.E. & Maley, F.M. (1985), Algorithms for routing and testing routability of planar VLSI-layouts, in: *Proceedings of the 17th Annual ACM Symposium on the Theory of Computing*, ACM, 1985, pp. 69–78. [31]

Lenstra, J.K. & Rinnooy Kan, A.H.G. (1979), Computational complexity of discrete optimization problems, *Annals of Discrete Mathematics* **4** (1979) 121–140. [12]

Li, C., McCormick, S.T. & Simchi-Levi, D. (1990), The complexity of finding two disjoint paths with min-max objective function, *Discrete Applied Mathematics* **26** (1990) 105–115. [65, 69]

Lovász, L. & Plummer, M.D. (1986), *Matching Theory*, Elsevier, Amsterdam, 1986. [15, 127]

Lynch, J.F. (1975), The equivalence of theorem proving and the interconnection problem, *ACM SIGDA Newsletters* **5** (1975) 31–65. [6, 29]

Mata, J.M. (1984), Solving system of linear equalities and inequalities efficiently, *Congressus Numerantium* **44** (1984) 251–260. [43]

Megiddo, N. (1983), Towards a genuinely polynomial algorithm for linear programming, *SIAM Journal on Computing* **12** (1983) 347–353. [43, 56]

Middendorf, M. & Pfeiffer (1993), On the complexity of the disjoint paths problem, *Combinatorica* **13** (1993) 97-107. [29]

Minty, G.J. (1960), Monotone networks, *Proceedings of the Royal Society of London* A **257** (1980) 194–212. [87]

Moore, E.F. (1959), The shortest path through a maze, in: *Proceedings of an International Symposium on the Theory of Switching, 2-5 April, Part II* [The Annals of the Computation Laboratory of Harvard University Volume XXX] (H. Aiken, ed.), Harvard University Press, 1959, pp. 285–292. [18, 42]

Nishizeki, T. & Chiba, N. (1988), *Planar Graphs: Theory and Algorithms*, North-Holland, Amsterdam, 1988. [23]

Orlin, J.B. (1993), A faster strongly polynomial minimum cost flow algorithm, *Operations Research* **41** 338–350. [58]

Oxley, J.G. (1992), *Matroid Theory*, Oxford University Press, Oxford, 1992. [136, 137, 147]

Padberg, M.W. & Rao, M.R. (1982), Odd minimum cut-sets and b-matchings, *Mathematics of Operations Research* **7** (1982) 67–80. [125, 126]

Papadimitriou, C.H. (1981), On the complexity of integer programming, *Journal of the Association for Computing Machinery* **28** (1981) 765–768. [61]

Papadimitriou, C.H. (1994), *Computational Complexity*, Addison-Wesley, Reading, MA, 1994. [12]

Papadimitriou, C.H. & Steiglitz, K. (1982), *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, N.J., 1982. [12, 16, 61]

Perl, Y. & Shiloach, Y. (1978), Finding two disjoint paths between two pairs of vertices in a graph, *Journal of the Association for Computing Machinery* **25** (1978) 1–9. [5, 23, 24, 65]

Pinter, R.Y. (1983), River routing: methodology and analysis, in: *Third Caltech Conference on Very Large Scale Integration*, Springer-Verlag, 1983, pp. 141–163. [29]

Recski, A. (1989), *Matroid theory and its applications in electrical network theory and in statics*, Springer-Verlag, Berlin, 1989. [117]

Reif, J.H. (1983), Minimum $s - t$ cut of a planar undirected network in $O(n \log^2 n)$ time, *SIAM Journal on Computing* **12** (1983) 71–81. [134]

Rockafellar, R.T. (1984), *Network and Monotropic Optimization*, John Wiley & Sons, New York, 1984. [86, 87]

Schrijver, A. (1978), The dependence of some logical axioms on disjoint transversals and linked systems, *Colloquium Mathematicum* **39** (1978) 191–199. [40, 47]

Schrijver, A. (1986), *Theory of Linear and Integer Programming*, John Wiley & Sons, Chichester, 1986. [16, 61, 88, 104, 130, 138]

Schrijver, A. (1991), Disjoint homotopic paths and trees in a planar graph, *Discrete Computational Geometry* **6** (1991) 527–574. [6, 31, 39, 48]

Schrijver, A. (1993), A group-theoretical approach to disjoint paths in directed graphs, *CWI Quarterly* **6** (1993) 257–266. [5, 23]

Schrijver, A. (1994), Personal communication. [7, 82, 83]

Schrijver, A. & Steenbeek, A. (1994), Dienstregelingontwikkeling voor Railned, Rapport CADANS 1.0 (in Dutch), Centrum voor Wiskunde en Informatica, preprint. [7, 82]

Serafini, P. & Ukovich, W. (1989), A mathematical model for periodic scheduling problems, *SIAM Journal on Discrete Mathematics* **2** (1989) 550-581. [83, 86]

Seymour, P.D. (1979), Sums of circuits, in: *Graph Theory and Related Topics* (J.A. Bondy and U.S.R. Murty, eds.), Academic Press, New York, 1979, pp. 341–355. [104]

Seymour, P.D. (1980), Decomposition of regular matroids, *Journal of Combinatorial Theory* (B) **28** (1980) 305–359. [137]

Seymour, P.D. (1985), Minors of 3-connected matroids, *European Journal of Combinatorics* **6** (1985) 375–382. [137]

Sleator, D.D. & Tarjan, R.E. (1983), A data structure for dynamic trees, *Journal of Computer and System Sciences* **24** (1983) 362–391. [126, 133]

Stepanets, G.F. (1964), Bases of cycle-vectors with extremal properties in graphs (in Russian), *Uspekhi Matematicherskikh Nauk* **19** (1964) 171–175 [Abstract: *Mathematical Reviews* **30** (1965) 484]. [92]

Stockmeyer, L.J. (1973), Planar 3-colorability is $\mathcal{NP}$-complete, SIGACT News **5** (1973) 19–25. [23]

Sysło, M.M. (1981), An efficient cycle vector space algorithm for listing all cycles of a planar graph, *SIAM Journal on Computing* **10** (1981) 797–808. [92]

Tardos, É. (1986), A strongly polynomial algorithm to solve combinatorial linear programs, *Operations Research* **34** (1986) 250–256. [139]

Tarjan, R.E. (1972), Depth-first search and linear graph algorithms, *SIAM Journal on Computing* **1** (1972) 146–160. [17, 45, 148]

Tarjan, R.E. (1978), Complexity of combinatorial algorithms, *SIAM Review* **20** (1978) 457-491. [12]

Tarjan, R.E. (1983), *Data Structures and Network Algorithms*, SIAM, Philadelphia, PA, 1983. [15, 18, 45, 125]

Truemper, K. (1987), Max-flow min-cut matroids: polynomial testing and polynomial algorithms for maximum flow and shortest routes, *Mathematics of Operations Research* **12** (1987) 72–96. [136, 139, 140, 142]

Truemper, K. (1992), *Matroid Decomposition*, Academic Press, San Diego, 1992. [136, 137, 138, 139]

Tutte, W.T. (1960), An algorithm for determining whether a given binary matroid is graphic, *Proceedings of the American Mathematical Society* **11** (1960) 905–917. [138]

Tutte, W.T (1968), *Connectivity in Graphs*, University of Toronto Press, Toronto, 1968. [145]

Wagner, R.A. (1976), A shortest path algorithm for edge-sparse graphs, *Journal of the Association for Computing Machinery* **23** (1976) 50–57. [18]

Warshall, S. (1962), A theorem on boolean matrices, *Journal of the Association for Computing Machinery* **9** (1962) 11–12. [19]

Welsh, D.J.A. (1976), *Matroid Theory*, Academic Press, London, 1976. [136]

Whiting, P.D. & Hillier, J.A. (1960), A method for finding the shortest route through a road network, *Operations Research Quartely* **11** (1960) 37–40. [18]

Wilson, R.J. (1972), *Introduction to Graph Theory*, Oliver and Boyd, Edinburgh, 1972. [3rd edition: Logman, Harlow (Essex, England), 1985]. [9]

Wolsey, L.A. (1974), A view of shortest route methods in integer programming, *Cahiers du Centre d'Études de Recherche Opérationelle* **16** (1974) 317–335. [61]

Zykov, A.A. (1969), *Theory of Finite Graphs*, Nauka, Novosibirsk, 1969 (in Russian). [92]

Index

1-sum 137

2-connected graph 10
2-satisfiability problem 40
2-sum 137

3-Satisfiability problem 66
3-sum 137

acceptable choice 63
acyclic graph 11
adjacent vertices 9
algorithm 13
all-pairs minimum cut problem 127
all-pairs shortest path problem . 17
arc 11

basis of circuits 10
basis of circuits of a digraph ... 12
basis of cuts 10
basis of cuts of a digraph 12
bidirected graph 48
binary matroid 135
block 10
bond 145

child marker 146
chord 111
circuit 8
circuit of a matroid 135
circuit space of a digraph 12
circuit space of a graph 10
circuit space of a matroid 135
clique 119
clique problem 119
cocycle 10
cographic matroid 135
combinatorial problem 13
components of a graph 10
components of a sum 137
concatenation of paths 10
connected graph 10
contraction of an edge 10
convex hull 8
cotree 10
cross-free cocycles 128
cross-free sets 128
cross-free shortest cut basis ... 128
crossing paths 33
current 114
cut 10
cut inequality 102
cut space of a digraph 12
cut space of a graph 10
cut-equivalent tree 127
cycle 9
cycle in a bidirected graph 49
cycle of a matroid 135
cyclomatic number 118

decision problem 13
digraph 11
directed cycle 11
disjoint homotopic paths problem 31
doubly-odd cycle 49
doubly-odd formula 48
dual graph 133

edge incidence 9
edge induced subgraph 9
edge of a graph 9
electrical network 114
endpoint of a path 9
endpoints 9
even cycle 12

feasible choice 62
feasible differential problem 86
Feasible Differential Theorem .. 87
flow equivalent tree 127
forest 10
fundamental circuit 10
fundamental cut 11

graph minor 10
graphic matroid 135
Gomory-Hu family of cuts 128
Gomory-Hu tree 127

Halin graph 106
Hamiltonian circuit 10
Hamiltonian circuit problem ... 13
head of an arc 11
homotopic paths 31

incidence matrix 11
incidence vector 7, 101
independent circuits 93, 124
independent signs 93, 124
independent vertex sets 124
instance 13
integer knapsack problem 61
internal vertices of a path 9
intractable problem 15
isomorphic matroids 137

k-colourable 23
k-colouring 23
k-connected matroid 137
k-cut 120
k-cut problem 120
k-separation 145
k-sum decomposition 137
Korach cycle 49

l_∞-norm 8
length-bounded disjoint paths .. 65
linear-time algorithm 14
logarithmic cost measure 14
longest path problem 26
loop 9
loop of a matroid 137
lower span 87

matroid contraction 137
matroid deletion 137
matroid intersection algorithm 138
maximal element 55
maximum flow minimum cut problem 125
merged graph 146
min-max problem 6, 60
min-sum problem 6, 60
minimal element 55
minimum $\{r, s\}$-cut problem .. 125
minimum spanning tree problem 13
minimum weighted T-cut problem 124
minimum weighted perfect matching problem 95
minor 137
minor-closed 122
multiterminal cut problem 120

negative-length cycle 17
network graph 114
$\mathcal{NP}$-complete in the strong sense 16
$\mathcal{NP}$-complete problem 15
$\mathcal{NP}$-hard problem 15
nullspace 135

odd cycle 12
odd edges 12
One-in-three 3SAT 69
optimization problem 13

parallel edges 9
parent marker 146
path 9
path in a bidirected graph 48
path packing polytope 62
period of an event 85
periodic event 85
periodic scheduling problem ... 85
perturbed lengths 121, 147
planar 3-colourability 23
planar graph 9
polynomial equivalence 136
polynomial-time algorithm 14

possible choice 62
potential 85
potential function 41
potential difference 86
pre-Leontief matrix 55
problem 13
problem size 14
proper minor 137
pseudo-polynomial algorithm .. 16

random access machine (RAM) 13
reachability 11
reachability problem 13
regular matroid 137, 138
representation matrix 135
resistance 115
rim edge 107
rim vertex 107

search problem 13
self-crossing path 33
semisimple cycle 49
shortest circuit basis ... 6, 82, 135
shortest circuit basis problem .. 92
shortest circuit of a matroid .. 136
shortest cut basis problem 117, 123
shortest disjoint path problem . 61
shortest f-circuit problem 136
shortest fundamental circuit basis problem 93
shortest fundamental cut basis problem 123
shortest nonhomotopic path problem 36
shortest odd circuit problem ... 94
shortest path 17
shortest-path tree 17
shortest $\mathcal{U}$-basis problem 117
sign 12
signed graph 12
signing 138
simple cycle of a bidirected graph 49
simple path 9
single-source shortest path problem 17
size of an integer 14
space storage 14
span interval 85
spanning cotree 10
spanning tree 10
spoke 107
star 132
strong component 11
strongly polynomial-time algorithm 16
strongly-connected component . 11
subgraph 9
symmetric difference 9

T-cut 124
t-decomposition 145
tail of an arc 11
tight cut 108
topological ordering 11
totally unimodular matrix 138
tractable problem 15
train timetabling problem 82
tree 10
tree-representation 130
trivial inequality 102

$\mathcal{U}$-basis 118
$\mathcal{U}$-path 117
$\mathcal{U}$-space 117
underlying graph 11
undirected graph 9
uniform cost measure 14
upper span 87

vertex 9
vertex induced subgraph 9
vertex of a graph 9
voltage 114

wheel 62, 107
work force scheduling 27
worst-case analysis 14

Samenvatting

Toepassingen van Kortste Pad-Methoden

Het kortste pad-probleem is een van de meest elementaire en fundamentele problemen in de theorie van transport- en communicatienetwerken. Veel algoritmen voor netwerk- en andere combinatorische optimaliseringsproblemen gebruiken kortste pad-algoritmen als belangrijke subroutines.

Het thema van dit proefschrift is de toepassing van kortste pad-methoden op een aantal praktische problemen, met bijzondere aandacht voor de analyse van de complexiteit van de problemen en de efficiëntie van de methoden.

Het eerste hoofdstuk beschrijft zo'n praktische probleem dat met kortste pad-methoden opgelost kan worden: de toewijzing van vliegtuigen van een luchtvaartmaatschappij aan vluchten. Hierbij wordt geëist dat bepaalde, voorgeschreven, vliegtuigen op een voorgeschreven dag op de thuisbasis zijn voor controle en onderhoud.

VLSI-routering, een van de aspecten van het ontwerpen van computerchips, kan geformuleerd worden als het vinden van disjunkte paden in een planaire graaf. Dit laatste is in het algemeen moeilijk ('$\mathcal{NP}$-moeilijk'). Echter, voor sommige speciale gevallen zijn efficiënte methoden (d.w.z. met polynomiaalbegrensde looptijd) beschikbaar. Zo gaf Schrijver [1991] een efficiënte methode die in een planaire graaf puntdisjunkte paden vindt die op een vooraf 'geschetste' wijze door de graaf lopen. (In de context van VLSI-routering is zo'n 'schets vooraf' een specificatie van de wijze—in topologische zin—waarop de verbindingen in de te ontwerpen chip, ten opzichte van de modules, door de chip moeten lopen.)

In Schrijver's algoritme, beschreven in Hoofdstuk 2, spelen kortste pad-methoden een belangrijke rol. Een van de stappen van dit algoritme behelst het vinden van een geheeltallige oplossing van een stelsel $Ax \leq b$ van lineaire ongelijkheden, waarbij A een $\{0, \pm 1\}$-matrix is met hooguit twee niet-nul coëfficiënten per rij. Schrijver toonde aan dat indien zo'n oplossing bestaat,

deze door Fourier-Motzkin eliminatie in polynomiale tijd gevonden kan worden. In Hoofdstuk 3 laten we zien dat dit sneller kan door het bepalen van kortste paden.

In Hoofdstuk 4, beschouwen we het probleem

$$\max\{c^T x \mid Ax \leq b,\ x \text{ geheeltallig}\},$$

waarbij A een matrix is als hierboven. Dit probleem is $\mathcal{NP}$-moeilijk, maar we tonen aan dat, indien c louter niet-negatieve coëfficiënten heeft en bovendien in elke rij van A hooguit een coëfficiënt positief is, dit maximum gevonden kan worden door middel van de oplossing van een kortste pad-probleem.

Een van de aspecten van de bestudering van de betrouwbaarheid van een communicatienetwerk is het vinden van veel disjunkte verbindingen tussen twee stations zodat een zeker kostencriterium geminimaliseerd wordt. Voorkomende kostencriteria zijn: de som van de transmissietijden van de gekozen verbindingen en de tijd waarbinnen de laatste transmissie voltooid wordt. Met dergelijke criteria is het probleem veel moeilijker dan zonder: zelfs in gevallen waarin disjunkte verbindingen in polynomiale tijd gevonden kunnen worden, kunnen de optimaliseringscriteria $\mathcal{NP}$-moeilijke problemen opleveren. In Hoofdstuk 5 laten we zien dat in speciale gevallen (waarin steeds het netwerk planair is) deze problemen met behulp van dynamische programmering in polynomiale tijd opgelost kunnen worden. Dynamische programmering kan gezien worden als het vinden van een kortste pad in een (grote) acyclische deelgraaf.

Als motivering voor het onderwerp van Hoofdstuk 7, wordt in Hoofdstuk 6 het probleem beschreven van het maken van een periodieke treindienstregeling die voldoet aan zekere randvoorwaarden. Schrijver & Steenbeek [1994] ontwierpen een methode hiervoor, welke onder meer gebruik maakt van een basis van circuits in een bepaalde graaf. De snelheid van de methode hangt onder meer af van de totale lengte van circuits in de gebruikte basis. Het onderwerp van Hoofdstuk 7 is het vinden van een circuit basis met minimale totale lengte. We beschrijven hiervoor twee methoden, een nieuwe en een van de hand van Horton [1987]. Beide gebruiken kortste pad-methoden als belangrijke subroutines. De methode geeft een verbeterde looptijd.

Tevens bestuderen we de relatie tussen circuit bases en oneven circuits in 'gesigneerde' grafen, alsmede een met circuit bases gerelateerd polyeder. We besluiten het hoofdstuk met een toepassing van kortste circuit bases in de analyse van electrische netwerken.

Circuits en sneden in een graaf kunnen beschouwd worden als 'duale' objecten. In Hoofdstuk 8 bestuderen we kortste bases van sneden in een graaf. De resultaten zijn duaal aan die van Hoofdstuk 7, dus de relatie met kortste paden is hier via dualiteit.

Zowel de collecties van circuits in een graaf als de collectie van sneden vormen een 'binaire matroide', d.w.z. een lineaire ruimte over GF(2).

In Hoofdstuk 9 worden de resultaten van de twee voorafgaande hoofdstukken gegeneraliseerd naar meer algemene binaire matroiden.

De resultaten in dit proefschrift werden gedeeltelijk behaald in samenwerking met A.M.H. Gerards (Hoofdstuk 8 en 9), H. van der Holst (Hoofdstuk 5) en A. Schrijver (Hoofdstuk 3, 5, 7, 8 en 9).