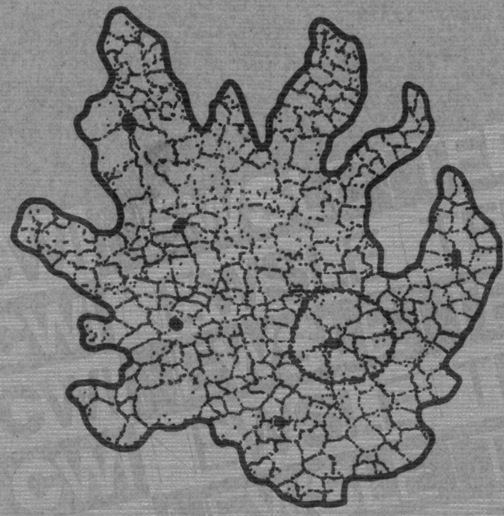
CWI NEWSLETTER

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

Quarterly, Issue no. 11
June 1986



CWI NEWSLETTER

Number 11, June 1986

The CWI Newsletter is published quarterly by the Centre for Mathematics and Computer Science (Centrum voor Wiskunde en Informatica), Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. The Newsletter will report on activities being conducted at the Centre and will also contain articles of general interest in the fields of Mathematics and Computer Science, including book reviews and mathematical entertainment. The editors encourage persons outside and in the Centre to contribute to the Newsletter. Normal referee procedures will apply to all articles submitted.

The Newsletter is available free of charge to all interested persons. The Newsletter is available to libraries on an exchange basis.

All correspondence should be addressed to: *The CWI Newsletter, Centre for Mathematics and Computer Science, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands.*

ISSN 0168-826x

Contents

CWI

Centre for Mathematics and Computer Science

Centrum voor Wiskunde en Informatica

On the History of Numerical Methods for Volterra Integral Equations

Hermann Brunner
Institut de Mathématiques Université de Fribourg
CH-1700 Fribourg, Switzerland

1. Introduction

In this note we present a short historical account of the development of numerical methods for Volterra integral equations of the second kind, with the main part of the paper (Section 3) covering the period between about 1920 and the early 1960s. In order to see this development in its proper context we begin with a section in which we sketch the origins and some of the classical theory of Volterra integral equations, and we conclude by subsequently describing the principal areas of current research and a number of recent automatic computer codes.

The paper is concerned with linear (one-dimensional) Volterra integral equations of the second kind, i.e., equations in the (continuous) function y of the form

$$y(t) = g(t)+\int_0^t K(t,s)y(s)ds, \quad t\in I:=[0,T] , \tag{1.1}$$

and

$$y(t) = g(t)+\int_0^t (t-s)^{-\alpha}y(s)ds, \quad t\in I, \ 0<\alpha<1 . \tag{1.2}$$

Their nonlinear counterparts are

$$y(t) = g(t)+\int_0^t k(t,s,y(s))ds, \quad t\in I , \tag{1.3}$$

and

$$y(t) = g(t)+\int_0^t (t-s)^{-\alpha}k(t,s,y(s))ds, \quad t\in I, \ 0<\alpha<1 . \tag{1.4}$$

It will be assumed that the kernels $K(t,s)$ and $k(t,s,y)$ are given continuous function of their respective variables and (in the nonlinear case) are such that there exists a unique solution $y \in C(I)$ whenever the given function g is in $C(I)$. (Generalizations to, e.g., the L_2-setting are, of course, possible but will not be considered here.)

2. A short history of Volterra integral equations

The classical papers of ABEL [1], [2] and of VOLTERRA [89] deal with the 'inversion of definite integrals': if g and G are given functions of one and two variables, respectively, find a (continuous) function y satisfying the first-kind integral equation

$$\int_0^t (t-s)^{-\alpha} G(t,s) y(s) ds = g(t), \quad t \in I, \ 0 \leqslant \alpha < 1 . \tag{2.1}$$

ABEL investigated the special case $G(t,s) \equiv 1$, $0 < \alpha < 1$, and derived the explicit inversion formula,

$$y(t) = \frac{\sin(\alpha\pi)}{\pi} \frac{d}{dt} \left(\int_0^t (t-s)^{\alpha-1} g(s) ds \right) . \tag{2.2}$$

He shows that equation (2.1), with $G(t,s) \equiv 1$, describes the problem of determining the equation of a curve in a vertical plane such that the time taken by a mass point to slide, under the influence of gravity, along this curve from a given positive height to the horizontal axis is equal to a prescribed (monotone) function of the height.

The general case was treated by VOLTERRA [89] in 1896: he showed, both for $\alpha = 0$ and for $\alpha \in (0,1)$, that if $G(t,t) \neq 0$ for all $t \in I$ and if g and G satisfy some obvious regularity conditions, then (2.1) can be rewritten as a second-kind integral equation (1.1) whose kernel is continuous on the domain $S := \{(t,s) : 0 \leqslant s \leqslant t \leqslant T\}$. Picard's method of successive approximations (proposed in his paper [72] of 1890) can then be employed; it leads, by means of the iterated kernels:

$$K_n(t,s) := \int_s^t K(t,v) K_{n-1}(v,s) dv \quad (n \geqslant 2), \ K_1(t,s) := K(t,s) ,$$

associated with $K(t,s)$ in (1.1), and the corresponding Neumann series:

$$R(t,s) := \sum_{n=1}^{\infty} K_n(t,s), \quad (t,s) \in S \tag{2.3}$$

(which, for $K \in C(S)$, converges absolutely and uniformly on S), to the 'inversion formula'

$$y(t) = g(t) + \int_0^t R(t,s) g(s) ds, \quad t \in I . \tag{2.4}$$

This inversion formula is no longer explicit, in the sense of (2.2), since the resolvent kernel $R(t,s)$ cannot, in general, be found analytically.

We mention in passing that one of the tools used in the above convergence analysis is Dirichlet's formula (dating from 1837; cf. [26]) which states that

$$\int_0^a dt \int_0^t \Phi(t,s)ds = \int_0^a ds \int_s^a \Phi(t,s)dt \ ,$$

provided Φ is a continuous function. (This result was generalized in 1908 by HURWITZ [49] to include integrands containing weakly singular terms like $t^{\lambda-1}(t-s)^{\mu-1}$, with $\lambda>0$, $\mu>0$.)

Particular cases of the second-kind integral equation (1.1) occur already in the papers by POISSON [73] of 1826, where the kernel $K(t,s)$ is of convolution type (i.e., depends only on the difference $t-s$ of its arguments t and s), and by LIOUVILLE [58] of 1837. LIOUVILLE seems to have been the first to employ the idea of successive approximations in an integral equation, thus anticipating Picard's suggestion by some fifty years: in loc.cit. he applied it to the integral equation he had obtained by rewriting the initial-value problem for a second-order differential equation, and he so established the uniform convergence of the resulting sequence of approximants. (His idea was subsequently extended to ordinary linear differential equations of arbitrary order by CAQUÉ in 1864; cf. [14] for bibliographical details.) As far as the general integral equation (1.1) is concerned, one finds the approach used by VOLTERRA already in the thesis by LE ROUX in 1894 (published as [56] a year later); however, LE ROUX did not investigate the uniform convergence of the resulting Neumann series.

By the turn of the century the classical quantitative theory of linear Volterra integral equations with regular kernels had essentially been established. Later work on second-kind integral equations by EVANS [31] in 1910/11 concerning various types of singular kernels, by ANDREOLI [3] in 1914 concerning equations whose upper limit of integration, t, is replaced by some function $\Phi(t)$, and by LALESCO, SCHMIDT, and others at about 1908 (see [22], [42], [25] for details) was already overshadowed by the fundamental work by FREDHOLM in 1900, 1903 ([33]) and by HILBERT in 1904-1910 ([43]). The latter work on second-kind integral equations with fixed limits of integration marks the birth of functional analysis. (Compare the recent studies by MONNA [68] and by DIEUDONNÉ [25]; see also [30].)

3. EARLY NUMERICAL METHODS

The idea of replacing the integral in (1.1), with $t=t_n:=nh$ $(n=1,...,N;\ Nh=T)$, by a finite sum (i.e., by some quadrature formula), thus obtaining, in a recursive way, approximations $\{y_n\}$ to the exact values $\{y(t_n)\}$, was introduced by VOLTERRA in [89, pp. 219-220] and, more explicitly, in [90, pp. 40-45]. Setting

$$y_n = g(t_n)+h \cdot \sum_{j=0}^{n-1} K(t_n,t_j)y_j, \quad n=0,...,N \tag{3.1}$$

he obtained a linear system in $\mathbb{R}^{N+1}$ for $u_N := (y_0,y_1,...,y_N)^T$. (Note that the right hand side corresponds to a particular Riemann sum, in which the values of the integrand are taken at the left endpoints of the subintervals $[t_j, t_{j+1}]$, $j=0,...,n-1$.) The system (3.1) has the form $(I_N - hA_N)u_N = g_N$. Here, I_N is the identity matrix, A_N is a strictly lower triangular matrix whose nontrivial elements are $K(t_n, t_j)$ $(0 \leqslant j < n \leqslant N)$, and $g_N := (g(t_0),...,g(t_N))^T$. Due to the special choice of the quadrature approximation the matrix $I_N - hA_N$ is always nonsingular (for more general quadrature formulas this will hold only for sufficiently large values of N), and hence (3.1) is uniquely solvable. VOLTERRA employed this approach not for the actual numerical solution of (1.1) but to establish, by 'passing from finiteness to infinity' (as he called it), the identities

$$\begin{aligned} R(t,s) &= K(t,s) + \int_s^t K(t,v)R(v,s)dv \\ &= K(t,s) + \int_s^t R(t,v)K(v,s)dv, \quad (t,s) \in S\ , \end{aligned}$$

(now generally referred to as Fredholm identities) between the kernel $K(t,s)$, of (1.1) and the corresponding resolvent kernel $R(t,s)$ introduced in (2.3).

3.1 Related fields

Before we start discussing the early contributions to the numerical solution of integral equations, beginning with WHITTAKER' s paper [92] of 1918, we shall recall briefly what was known at that time in two fields closely related to numerical analysis, namely numerical integration (or quadrature) and the numerical solution of initial-value problems for ordinary differential equations. (The reader is referred to GOLDSTINE [36], GAUTSCHI [35], MILNE [67], and to the forthcoming monograph [40] by HAIRER, NØRSETT and WANNER for bibliographical and historical details.)

Numerical quadrature. The origins of numerical quadrature date back to the work of CAVALIERI (1639), GREGORY (1670), NEWTON (1676), COTES (1722), MACLAURIN (1742), SIMPSON (1743), and EULER (1755). The results of GAUSS (1814) on more general quadrature formulas were extended by JACOBI (1826) (who based his theory on the theory of orthogonal polynomials), and by CHRISTOFFEL (1852). The work of LOBATTO and RADAU on quadrature formulas possessing a certain number of prescribed abscissas (either both, or one of the endpoints of the interval of integration) dates from 1852 and 1880 respectively. Finally, the classical result on the integral representation of the error of a given quadrature formula, i.e. Peano's kernel theorem, was published in 1914.

Among the classical quadrature formulas it was the one known as Gregory's rule which, as will be seen below, played initially the dominant role in the numerical solution of Volterra integral equations Gregory's rule in an extension of the trapezoidal rule, it has the form

$$\int_0^{t_n} f(s)ds \approx h\left[\frac{1}{2}f(t_0)+f(t_1)+...+f(t_{n-1})+\frac{1}{2}f(t_n)\right] - \qquad (3.2)$$

$$- h\cdot\sum_{i=1}^{q} c_i\left[\nabla^i f(t_n)+(-1)^i\Delta^i f(t_o)\right],$$

where $t_k := t_0 + kh$ $(h>0)$, $t_0 := 0$, $\Delta^0 f(t_k) := f(t_k)$, $\Delta f(t_k) := f(t_{k+1}) - f(t_k)$, $\Delta^i := \Delta(\Delta^{i-1})$ (with analogous definitions for the backward difference operator $\nabla f(t_k) := f(t_k) - f(t_{k-1})$), and where the first few coefficients c_i in the end corrections are given by $c_1 = 1/12$, $c_1 = 1/24$, $c_3 = 19/720$, $c_4 = 3/360,...$, q is a given integer. Notice that the case $q=0$ corresponds to the trapezoidal rule. The above quadrature formula (3.2) is closely related to the Euler-MacLaurin summation formula (established some seventy years after Gregory's formula),

$$\int_0^{t_n} f(s)ds = h\left[\frac{1}{2}f(t_0)+f(t_1)+...+f(t_{n-1})+\frac{1}{2}f(t_n)\right] -$$

$$-\sum_{k=1}^{m}\frac{h^{2k}B_{2k}}{(2k)!}\left[f^{(2k-1)}(t_n)-f^{(2k-1)}(t_0)\right] + R_m(f),$$

where $R_m(f) := -t_n h^{2m+2} B_{2m+2} f^{(2m+1)}(\xi)/(2m+2)!$ $(\xi\in[t_0,t_n])$; here, the B_j are the Bernoulli numbers (i.e., the coefficients of $t^j/j!$ in the power series expansion of $t/(e^t-1)$). Gregory's rule is obtained by using appropriate finite-difference approximations to the derivatives of f at the endpoints t_0 and t_n, followed by suitable truncation (compare also KRYLOV [54, pp. 35-38]).

Initial-value problems. In the development of numerical methods for the initial-value problem $y' = f(t,y)$, $y(t_0) = y_0$, an idea first encountered in Euler's work (1768) was used by GAUCHY in 1840 to derive a viable algorithm (now generally known as Euler's method), $y_{n+1} := y_n + hf(t_n,y_n)$ $(n \geqslant 0)$. Among the successors of this method are the one-step methods of RUNGE (1895), HEUN (1900), and KUTTA (1901) one of whose explicit four-stage, fourth order methods was, until fairly recently, simply referred to as *the* Runge-Kutta method. The (explicit) linear multistep methods known as Adams-Bashforth methods which were introduced by BASHFORTH and ADAMS in 1883 can also be considered as successors to Euler's method. The analogous implicit linear multistep methods (the methods of Adams-Moulton) originate from the work of MOULTON in the 1920s. Nyström's method for approximating the solution to the initial-value problem for a second-order differential equation dates from 1926. Except for some earlier surveys, the books by COLLATZ [21] and by MILNE [67] (whose first editions were published in 1951 and 1953 respectively) represent the first comprehensive accounts of numerical methods for ordinary differential equations.

Returning to Volterra integral equations and to the paper [92] by WHITTAKER of 1918, we observe that his methods for equations of the form (1.1) with convolution kernel $K(t,s)=a(t-s)$ do not yet reflect the fact that (1.1) may be viewed as a generalization of the initial-value problem for an ordinary differential equation, for whose numerical solution one might try to use suitable analogues of certain known methods for ordinary differential equations. Whittaker's first two methods are based on the assumption that the kernel $a(z)$ is given in the form of a numerical table. Using De Prony's method of 1795, he approximates $a(z)$ by an interpolant which is a linear combination of exponential functions; its construction involves the solution of a certain nonlinear algebraic equation. The approximation $u(t)$ to the exact solution is then of the form

$$u(t) = g(t)+\int_0^t r(t-s)g(s)ds, \quad t\in I ,$$

where $r(z)$ (which may be regarded as an approximation to the resolvent kernel associated with the kernel $a(z)$; cf. (2.4)) is again a linear combination of exponential functions whose exponents are given by the roots of the above nonlinear equation and whose coefficients depend on these roots as well as on the exponents occurring in the interpolant of $a(z)$. In the second method the interpolant is a polynomial, while in the third method it is assumed that one knows the Taylor expansion of $a(z)$; this then permits the computation of the Taylor expansion of the resolvent kernel. In both cases, the exact solution is approximated by an expression of the form (3.2).

3.2. The methods of Prasad

Multistep methods. The methods proposed by PRASAD [75] in 1924 are the true ancestors of most of the present-day numerical methods for (1.1) and (1.3); moreover, they are applicable not only to integral equations with convolution kernels but to general equations as well. His linear multistep method is based on the Gregory rule (3.2); for (1.3) it thus assumes the form

$$y_n = g(t_n)+h\sum_{j=0}^{n} w_{n,j}k(t_n,t_j,y_j), \quad n = q+1,...,N , \tag{3.3}$$

where $t_n := nh$ $(n=0,...,N;\ Nh=T)$, and where the weights $\{w_{n,j}\}$ are easily obtained from the coefficients characterizing Gregory's rule (3.2). PRASAD employs the value $q=4$; in addition to $y_0=g(0)$ he thus needs the values y_1,y_2,y_3,y_4 to start the recursion (3.3). These starting values may be obtained by means of the trapezoidal rule and Simpson's rule, possibly using smaller initial sub-intervals in order to attain sufficiently accurate approximations.

Runge-Kutta methods. In order to avoid methods that depend on starting values, PRASAD shows how the ideas of RUNGE, KUTTA and others can be adapted to generate approximations at $t_n = nh$ to the solution of (1.3). His starting point is one of the explicit four-stage, fourth-order methods introduced by KUTTA in 1901: in current notation (i.e., in terms of the so-called Butcher array: see, e.g., [40]) this method is characterized by

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} := \begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 \\ \frac{2}{3} & -\frac{1}{3} & 1 & 0 & 0 \\ 1 & 1 & -1 & 1 & 0 \\ \hline & \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{array} .$$

Hence, for $t \in [t_n, t_{n+1}]$ the integral equation (1.3) is discretized by setting

$$y_{n+1} = \tilde{F}_n(t_{n+1}) + h \cdot \sum_{i=1}^{m} b_i k(t_{n+1}, t_n + c_i h, Y_{n,i}) , \qquad (3.4a)$$

where the $Y_{n,i}$ are obtained from

$$Y_{n,i} = \tilde{F}_n(t_n + c_i h) + h \cdot \sum_{j=1}^{i-1} a_{i,j} k(t_n + c_i h, t_n + c_j h, Y_{n,j}) , \qquad (3.4b)$$

$$(j = 1, \ldots, m) .$$

Here, $m = 4$, and $\tilde{F}_n(t)$ denotes a suitable approximation to the lag term

$$F_n(t) := g(t) + \int_0^{t_n} k(t, s, y(s)) ds, \quad t \in [t_n, t_{n+1}] , \qquad (3.5)$$

of the equation (1.3), and the numbers b_i, c_i, and $a_{i,j}$ are the elements of the vectors b, c and of the matrix A, respectively, in the above array. PRASAD however, dismisses Runge-Kutta methods of the form (3.4) as being 'not so good as' the method based on Gregory's rule; they found a renewed interest only some thirty years later.

We note in passing the the first 'practical' application of a method of the form (3.3) (involving Gregory's rule (3.2) with $q = 0$) seems to occur in the book [20] by CARSON (pp. 145-146) in 1926; the method is employed to solve numerically a linear Volterra integral equation with convolution kernel found in the theory of electric circuits.

3.3. Convergence analysis

The first convergence analysis for quadrature methods (3.3) applied to the linear equation (1.1), exhibiting the relation between the errors of the underlying quadrature formula and the resulting order of the approximation error $e_n := y(t_n) - y_n$ of the method, was given by MIKELADZE [65] in 1935. The main tool in his analysis is a discrete version of Gronwall's inequality,

$$z_n \leq hC_1 \cdot \sum_{i=0}^{n-1} z_i + C_2, \quad n = 0, \ldots, N \ ,$$

with $z_i \geq 0$, $C_1 > 0$, $C_2 > 0$ (see also below, Section 4). The methods studied by MIKELADZE are essentially those based on the various Gregory rules. The author's motivation for considering these methods lies in the numerical solution of higher-order linear differential equations: he suggests their being rewritten as Volterra integral equations of the second kind. An analogous idea is also used for second-order linear partial differential equations, here, the resulting integral equations contain double integrals.

The paper by KRYLOV [54] of 1949 deals also with the quadrature method (3.3) employing the Gregory rule (in the Russian literature this rule is often called the Euler-Laplace formula). Moreover, KRYLOV introduces block methods for the simultaneous computation of the starting values y_1, y_2 (if $q=2$), or y_1, y_2, y_3, y_4 (if $q=4$) needed in (3.3). These starting methods are obtained by choosing sets of q quadrature formulas of the same length ($q+1$ abscissas) and with the same degree of precision. (Compare also Wolkenfelt's thesis and his paper [94] for these and other starting methods.) These block methods involve kernel values $K(t,s)$ or $k(t,s,y)$ for $s > t$ which have to be found by a suitable extrapolation procedure; moreover, a linear (or, in the case (1.3), a nonlinear) system in $\mathbb{R}^2$ or in $\mathbb{R}^4$ has to be solved. Although these starting methods are chosen so that their orders of accuracy are consistent with the orders of accuracy of Gregory rules underlying (3.3), there is no convergence analysis. MIKELADZE' s paper [65] of 1951 takes up the ideas of Krylov and suggests a number of marginal improvements in the implementation of the methods.

In the early 1950s, explicit Runge-Kutta methods of the form (3.4) were considered once more, namely in the paper [84] by SUYAMA and NAKAMORI. Here, the interest was focused on the derivation of the so-called order conditions which the parameters c_i, b_i, and $a_{i,j}$ in (3.4a), (3.4b) have to satisfy if the method is to have (local) order $p=4$. In other words, suppose that the lag term approximations $\tilde{F}_n$ in (3.4a) and (3.4b) are replaced by the *exact* lag term F_n introduced in (3.5) (i.e., the given integral equation (1.3) is solved *locally* on $[t_n, t_{n+1}]$, by assuming that $y(t)$ be known exactly on $[0, t_n]$); let the resulting approximation at $t = t_{n+1}$ be denoted by $\bar{y}_{n+1}$. What algebraic equations do the parameters c_i, b_i, and $a_{i,j}$ have to satisfy in order that $|y(t_{n+1}) - \bar{y}_{n+1}| \leq Ch^p$, where C is a constant not depending on h ? These order conditions are derived by Taylor expansion techniques (compare also [18] for a more elegant approach which is based on certain concepts from graph theory and which extends the analogous theory of Butcher for Runge-Kutta methods in ordinary differential equations). However, questions regarding the convergence or the practical implementation of (3.4) (including the problem of how to generate suitable lag term approximations $\tilde{F}_n$) are not touched upon.

The first systematic convergence analysis for explicit Runge-Kutta methods (3.4) is due to POUZET and may be found in his thesis of 1962 and in [74]; see also GAMONDI and ITALIANI [34] for closely related results. POUZET showed that if the lag term approximations $\tilde{F}_n$ are based on quadrature formulas characterized by the weights b_i and the abscissas t_k+c_ih $(i=1,...,4;\ k<n)$, and if the Runge-Kutta method (3.4) has local order $p=4$, then the approximation error $e_n := y(t_n)-y_n$ satisfies $\max_{(n)}|e_n| = \mathcal{O}(h^p)$, as $h\downarrow 0$, $Nh=const$. Analogous results hold for other, suitably chosen lag term approximations.

A different class of explicit Runge-Kutta methods for (1.3) was introduced by BEL'TYUKOV [9] in 1965. Here, the underlying arrays of parameters are no longer those given by a Runge-Kutta method for a differential equation. We now have, instead of (3.4),

$$y_{n+1} = \tilde{F}_n(t_{n+1})+h\cdot\sum_{i=1}^{m} b_i k(t_n+d_ih,\ t_n+c_ih,\ y_{n,i}),$$

with

$$y_{n,i} = \tilde{F}_n(t_n+c_ih)+h\sum_{j=1}^{i-1} a_{i,j}k(t_n+d_jh,\ t_n+c_jh,\ y_{n,j})\ (j=1,...,m).$$

BEL'TYUKOV analyzed the methods corresponding to $m\leqslant 3$ and satisfying the conditions $d_i\geqslant c_i$ for all values of i. Even though these methods require fewer kernel evaluations than the methods (3.4) studied by POUZET it turns out (cf. [18]) that the construction of higher-order methods (of order $p\geqslant 4$) is quite difficult; in particular, there does not exist an explicit Bel'tyukov method with $p=m=4$ (recall that, as shown by POUZET $p=m=4$ is possible for (3.4)).

As far as high-order methods of Runge-Kutta type are concerned, we note that SCHOEDON [80] in 1970 studied a class of such methods based on certain Hermite quadrature formulas.

Returning briefly to linear multistep methods of the form (3.3), we point out the paper [52] by JONES of 1961: this paper contains a detailed convergence analysis of the trapezoidal method when applied to second-kind integral equations with convolution kernels (or to systems of such equations). Later analyses of linear multistep methods (3.3) were largely influenced by the fundamental work of DAHLQUIST and HENRICI on linear multistep methods for ordinary differential equations (dating from the late 1950s and the early 1960s). The first papers to extend their theory to Volterra integral equations are due to SPOHN [83] (1965) and to KOBAYASI [53] (1966).

Up to the early 1960s, almost no attention had been paid to the numerical solution of Volterra integral equations (1.2) and (1.4) whose kernels contain a weak (integrable) singularity of the form $(t-s)^{-\alpha}$, with $0<\alpha<1$. PRASAD [75,

p. 58] briefly mentions the possibility of rewriting (1.2), with rational $\alpha = p/q$ ($p,q \in \mathbb{N}$; p and q coprime), as an equation with regular kernel; the underlying change of variable implies that the upper limit of integration in (1.1) now becomes $t^{1/q}$. However, he gives no further details. WAGNER [91] in 1954 seems to be the first source suggesting in detail a numerical method for (1.3) and (1.4). Using, and at the same time generalizing, ideas contained implicitly in the paper [48] of HUBER (1939), he employs continuous, piecewise quadratic polynomials to approximate the solution to the given integral equation; this approximation is determined by so-called collocation techniques. While the application of this method to Volterra integral equations arising in heat conduction problems is given particular attention, there is no analysis of its convergence properties.

The subsequent development of numerical methods for Volterra integral equations possessing weakly singular kernels was based mainly on the work of YOUNG [96] of 1954 on product integration techniques. We refer the reader to the relevant references in [59] and [15] for additional details.

3.4. Conclusion

When surveying the contributions to the numerical solution of second-kind Volterra integral equations up to about 1965 one is perhaps struck by the fact that, with the possible exception of WHITTAKER [92] and MIKELADZE [65], they all deal with specific examples of methods and that a more unified view is still very much lacking. It seems interesting to observe that there emerges a rather different picture if one looks at the early development of numerical methods for Fredholm integral equations of the second-kind,

$$y(t) = g(t) + \lambda \int_0^T K(t,s) y(s) ds, \quad t \in I:$$

here, the two earliest methods, Bateman's method ([7]) of 1922 and Nyströms method ([71]) of 1928 represent very general approaches to generating numerical approximations to the solutions of such equations.

We conclude this section by mentioning that early surveys of numerical methods for Volterra and Fredholm integral equations (containing most, but not all, of the methods described here) may be found in BERNIER [10], FOX and GOODWIN [32], MAYERS [63], and NOBLE [69]. In addition, see also [85] and the extensive bibliography [70] by NOBLE.

4. RECENT DEVELOPMENTS

For the sake of completeness we name a few references to recent work on the approximate solution of second-kind Volterra equations.

(i) A very general analysis of quadrature methods (3.3) for (1.1) and (1.3) was recently given by WOLKENFELT [94] (see also his thesis of 1981). Generalizations of such methods are discussed in WOLKENFELT [95] and in VAN DER HOUWEN and TE RIELE [46], [47]. Compare also BRUNNER and VAN DER HOUWEN [19, Ch. 3]. Fractional quadrature methods for equations with weakly singular kernels were introduced by LUBICH [62].

(ii) A particular class of implicit Runge-Kutta methods (3.4) was analyzed in detail by DE HOOG and WEISS [45]. The general Runge-Kutta theory for Volterra equations (1.3) with smooth kernels was established by BRUNNER, HAIRER and NØRSETT [18]; a comprehensive convergence analysis for general one-step methods (including Runge-Kutta methods of Pouzet and Bel'tyukov type) is due to HAIRER, LUBICH and NØRSETT [38]. LUBICH [61] extended the Runge-Kutta theory of [18] to weakly singular Volterra equations (1.4).

(iii) A recent account of collocation methods based on polynomial splines may be found in BRUNNER [14]; see also [16]. While these papers focus on Volterra equations with regular kernels, [15] and [17] are concerned with the problem of generating high-order approximations to (nonsmooth) solution of equations with weakly singular kernels. Compare also TE RIELE [78] where non-polynomial spline functions are employed to obtain such approximations.

(iv) Abstract convergence analysis (including numerous examples) of discretization methods for second-kind Volterra integral equations may be found in SCOTT [81] and in DIXON and MCKEE [28].

(v) There still exists only very few relatively general analyses of numerical stability of methods for second-kind Volterra equations. The two principal ones, dealing with convolution kernels, are those by LUBICH [60] (for quadrature methods) and by HAIRER and LUBICH [37] (for Runge-Kutta methods (3.4)). As regards equations with more general kernels, we refer to [19, Ch. 7]. The problem of numerical stability when solving weakly singular Volterra equations is still very much in the open; see, however, LUBICH [62].

(vi) The principal tool in the convergence analysis of numerical methods is the discrete Gronwall inequality,

$$z_n \leqslant C_1 h^{1-\alpha} \sum_{i=0}^{n-1} (n-i)^{-\alpha} z_i + C_2, \quad n=0,...,N, \quad 0 \leqslant \alpha < 1 ,$$

where $z_i \geqslant 0$, $C_1 > 0$, $C_2 > 0$, and $Nh \leqslant c < \infty$. As mentioned in Section 3, the first occurrence of such an inequality (with $\alpha = 0$) seems to be in MIKELADZE [65, p. 259]. Of the more recent contributions in this area we mention the ones by JONES [50] and, especially, by BEESACK [8]. The case $0 < \alpha < 1$ is treated in detail in MCKEE [64], DIXON and MCKEE [27], and SCOTT [81].

(vii) Recent surveys of numerical methods for Volterra integral equations may be found in TE RIELE [77], BRUNNER [13], and BAKER [5]. The proceedings [24], [6] and [41] provide a good indication as to the current activities in the numerical analysis of Volterra equations. The first comprehensive monograph on the numerical treatment of Volterra (and Fredholm) integral equations, BAKER [4], is of quite recent origin: it appeared in 1977. More recent treatises are LINZ [57] and BRUNNER and VAN DER HOUWEN [19].

5. AUTOMATIC COMPUTER CODES

The development of efficient and reliable software for second-kind Volterra integral equations is a very new activity (compare the comments in [30, p. 13]; see also DELVES [23] and the article by MILLER in [24, pp. 247-256]). At the time of writing, both the IMSL and NAG libraries did not contain any procedures for such equations. For historical reasons we mention the collection of ALGOL procedures [76] where one finds a number of non-automatic codes based on Pouzet's work on Runge-Kutta methods and Adams-type quadrature methods. In addition see also [88].

In the following we list some of the recently developed codes involving local or global error estimation and/or automatic stepsize change.

(i) The code of Bownds and Appelbaum [12] is based on kernel approximation techniques, the resulting integral equation is then equivalent to a system of (nonlinear) ordinary differential equations which are solved by a standard Adams or Runge-Kutta-Fehlberg code. See also the pertinent comments in [82] on the choice of the differential equation code if the system turns out to be stiff.

(ii) Codes using specific quadrature methods of the form (3.3) were written by LOGAN [59] (Simpson's method, with a block-by-block option); HOCK [44] (midpoint method, followed by extrapolation techniques); KUNKEL [55], WILLIAMS and MCKEE [93], and JONES [51] (predictor-corrector techniques). The only automatic code for weakly singular equations (1.4) with $\alpha = 1/2$ is due to LOGAN [59] (product Simpson's method, used in block-by-block mode).

(iii) The following codes employ Runge-Kutta type methods: TANFULLA and RIBIGHINI [86] (explicit, embedded Pouzet methods of orders 4 and 5); DUNCAN [29] (explicit, embedded 6-stage and 8-stage methods of Pouzet type and with orders 5 and 6); SCHLICHTE [79] (implicit method of DE HOOG and WEISS [45]); HAIRER, LUBICH and SCHLICHTE [39] (explicit 4-stage Pouzet method of order 4, combined with fast Fourier transform techniques; this method is devised for equations with convolution kernels); and BLOM and BRUNNER [11] (implicit Pouzet-type methods of variable orders, combined with discretized iterated collocation; the resulting local superconvergence properties are used to obtain error estimates). All of these codes are designed for Volterra integral equations possessing regular (bounded) kernels.

A more detailed description of the above codes may be bound in [19, Ch. 8].

REFERENCES

1. N.H. ABEL (1823). Solution de quelques problèmes à l'aide d'intégrales définies. *Oeuvres complètes I,* 11-27, Grøndahl & Søn, Christiania (1881).
2. N.H. ABEL (1826). Auflösung einer mechanischen Aufgabe. *Oeuvres complètes I,* 97-101, Grøndahl & Søn, Christiania (1881).
3. G. ANDREOLI (1914). Sulle equazioni integrali. *Rend. Palermo 37,* 76-112.

4. C.T.H. BAKER (1977). *The Numerical Treatment of Integral Equations*, Oxford University Press, Oxford.
5. C.T.H. BAKER (1982). An introduction to the numerical treatment of Volterra and Abel-type integral equations. P.R. TURNER (ed.). *Topics in Numerical Analysis, Lancaster 1981, Lecture Notes in Math. 965,* 1-38, Springer-Verlag, Berlin-Heidelberg-New York.
6. C.T.H. BAKER, G.F. MILLER (eds.) (1982). *Treatment of Integral Equations by Numerical Methods* (Durham 1982), Academic Press, London.
7. H. BATEMAN (1922). On the numerical solution of linear integral equations. *Proc. Roy. Soc. London (A) 100,* 441-449.
8. P.R. BEESACK (1975). *Gronwall Inequalities, Carleton Math. Lecture Notes 11,* Carleton University, Ottawa.
9. B.A. BEL'TYUKOV (1965). An analogue of the Runge-Kutta method for the solution of nonlinear integral equations of Volterra type. *Differential Equations 1,* 417-433.
10. J. BERNIER (1945). Les principales méthodes de résolution numérique des équations intégrales de Fredholm et de Volterra. *Ann. Radio électr. 1,* 311-318.
11. J.G. BLOM, H. BRUNNER (1985). *The Numerical Solution of Nonlinear Volterra Integral Equations of the Second Kind by Collocation and Iterated Collocation Methods,* Report NM-R8522, CWI, Amsterdam.
12. J.M. BOWNDS, L. APPELBAUM (1985). Algorithm 627: A FORTRAN subroutine for solving Volterra integral equations. *ACM Trans. Math. Software,* 58-65.
13. H. BRUNNER (1982). A survey of recent advances in the numerical treatment of Volterra integral and integro-differential equations. *J. Comput. Appl. Math. 8,* 213-229.
14. H. BRUNNER (1984). On the discretization of Volterra integral equations. *Nieuw Arch. Wisk. (4) 2,* 189-217.
15. H. BRUNNER (1984). The numerical solution of integral equations with weakly singular kernels. D.F. GRIFFITHS (ed.). *Lecture Notes in Math. 1066,* 50-71, Springer-Verlag, Berlin-Heidelberg-New York.
16. H. BRUNNER (1984). Iterated collocation methods and their discretizations for Volterra integral equations. *SIAM J. Numer. Anal. 21,* 1132-1145.
17. H. BRUNNER (1985). The numerical solution of weakly singular Volterra integral equations by collocation on graded meshes. *Math. Comp. 45,* 417-437.
18. H. BRUNNER, E. HAIRER, S.P. NØRSETT (1982). Runge-Kutta theory for Volterra integral equations of the second kind. *Math. Comp. 39,* 147-163.
19. H. BRUNNER, P.J. VAN DER HOUWEN (1986). *The Numerical Solution of Volterra Equations,* North-Holland *(CWI Monographs 3),* Amsterdam.
20. J.R. CARSON (1926). *Electric Circuit Theory and the Operational Calculus,* Chelsea Publ. Co., New York (reprinted 1953).
21. L. COLLATZ (1951). *Numerische Behandlung von Differentialgleichungen,* Springer-Verlag, Berlin-Göttingen-Heidelberg.
22. H.T. DAVIS (1926). The present status of integral equations. *Indiana*

University Studies XIII, Study No. 70, Bloomington.
23. L.M. DELVES (1978). Numerical software for integral equations. D. JACOBS (ed.). *Numerical Software,* 303-323, Academic Press. London.
24. L.M. DELVES, J. WALSH (eds.) (1974). *Numerical Solution of Integral Equations,* Clarendon Press., Oxford.
25. J. DIEUDONNÉ (1981). *History of Functional Analysis, Notas de Matemática (77),* North-Holland, Amsterdam.
26. P.G.L. DIRICHLET (1837). Sur les séries dont le terme général dépend de deux angles, et qui servent à exprimer des fonctions arbitraires entre des limites données. *J. Reine Angew. Math. 17,* 35-56.
27. J. DIXON, S. MCKEE (1983). *Singular Gronwall Inequalities,* Numerical Analysis Report NA/83/44, Hertford College, University of Oxford.
28. J. DIXON, S. MCKEE (1985). A unified approach to convergence analysis of discretization methods for Volterra-type equations. *IMA J. Numer. Anal. 5,* 41-57.
29. R.P. DUNCAN (1982). *A Runge-Kutta Method Using Variable Stepsizes for Volterra Integral Equations of the 2nd Kind,* Techn. Report 157/82, Dept. of Computer Science, University of Toronto.
30. D. ELLIOTT (1980). Integral equations - ninety years on. R.S. ANDERSSEN et al. (eds.). *Application and Numerical Solution of Integral Equations,* 1-20, Sijthoff & Noordhoff, Alphen aan den Rijn.
31. G.C. EVANS (1910/11). Volterra's integral equation of the second kind with discontinuous kernel. *Trans. Amer. Math. Soc. 11,* 393-413; *12,* 429-472.
32. L. FOX, E.T. GOODWIN (1953). The numerical solution of nonsingular linear integral equations. *Philos. Trans. Roy. Soc. London Ser. A 245,* 501-534.
33. I. FREDHOLM (1903). Sur une classe d'équations fonctionnelles. *Acta Math. 27,* 365-390.
34. G. GAMONDI, M. ITALIANI (1965). Sull'analisi numerica delle equazioni integrali di Volterra. *Calcolo 2,* 249-266.
35. W. GAUTSCHI (1981). A survey of Gauss-Christoffel quadrature formulae. P.L. BUTZER, F. FEHER (eds.). *E.B. Christoffel: The Influence of his Work on Mathematics and the Physical Sciences,* 72-147, Birkhäuser Verlag, Basel.
36. H.H. GOLDSTINE (1977). *A History of Numerical Analysis from the 16th through the 19th Century,* Springer-Verlag, Berlin-Heidelberg-New York.
37. E. HAIRER, CH. LUBICH (1984). On the stability of Volterra-Runge-Kutta methods. *SIAM J. Numer. Anal. 21,* 123-135.
38. E. HAIRER, CH. LUBICH, S.P. NØRSETT (1983). Order of convergence of one-step methods for Volterra integral equations of the second kind. *SIAM J. Numer. Anal. 20,* 569-579.
39. E. HAIRER, CH. LUBICH, M. SCHLICHTE (1983). Fast numerical solution of nonlinear Volterra convolution equations. *SIAM J. Sci. Statist. Comput. 6,* 532-541.
40. E. HAIRER, S.P. NØRSETT, G. WANNER (1986). *Numerical Ordinary*

Differential Equations (to appear).
41. G. HÄMMERLIN, K.H. HOFFMANN (eds.) (1985). *Constructive Methods for the Practical Treatment of Integral Equations (Oberwolfach 1984),* ISNM 73, Birkhäuser Verlag, Basel-Boston.
42. E. HELLINGER, O. TOEPLITZ (1927). Integralgleichungen und Gleichungen mit unendlichvielen Unbekannten. *Encyklopädie der Math. Wissenschaften II, 3,* 1335-1601. (Reprint: Chelsea Publ. Co., New York, 1953).
43. D. HILBERT (1912). *Grundzüge einer allgemeinen Theorie der linearen Integralgleichungen,* Teubner, Leipzig.
44. W. HOCK (1981). An extrapolation method with step size control for non-linear Volterra integral equations. *Numer. Math. 38,* 155-178.
45. F. DE HOOG, R. WEISS (1975). Implicit Runge-Kutta methods for second kind Volterra integral equations. *Numer. Math. 23,* 199-213.
46. P.J. VAN DER HOUWEN, H.J.J. TE RIELE (1982). Linear multistep methods for Volterra integral equations of the second kind. In [6], 79-93.
47. P.J. VAN DER HOUWEN, H.J.J. TE RIELE (1985). Linear multistep methods for Volterra integral and integrodifferential equations. *Math. Comp. 45,* 439-461.
48. A. HUBER (1939). Eine Näherungsmethode zur Auflösung Volterrascher Integralgleichungen. *Monatsh. Phys. 47,* 240-246.
49. W.A. HURWITZ (1908). Note on certain iterated and multiple integrals. *Ann. of Math. 9,* 183-192.
50. G.S. JONES (1964). Fundamental inequalities for discrete and discontinuous functional equations. *J. Soc. Indust. Appl. Math. 12,* 43-57.
51. H.M. JONES (1985). A variable step variable order package for solving Volterra integral equations of the second kind (to appear in *ACM Trans. Math. Software*).
52. J.G. JONES (1961). On the numerical solution of convolution integral equations and systems of such equations *Math. Comp. 15,* 131-142.
53. M. KOBAYASI (1966). On numerical solution of the Volterra integral equations of the second kind by linear multistep methods. *Rep. Statist. Appl. Res. Un. Japan Sci. Engrs. 13,* 1-21.
54. V.I. KRYLOV (1949). Application of the Euler-Laplace formula to approximate solution of integral equations of Volterra type. *Trudy Mat. Inst. Steklov 28,* 33-72 (Russian).
55. P. KUNKEL (1982). *Ein adaptives Verfahren zur Lösung von Volterraschen Integralgleichungen zweiter Art,* Diplomarbeit, Institut für Angewandte Mathematik, Universität Heidelberg.
56. J. LE ROUX (1895). Sur les intégrales des équations linéares aux derivées partielles du second ordre à deux variables indépendantes. *Ann. Sci. Ecole Normale Supér. (3) 12,* 227-316 (cf. 243-246).
57. P. LINZ (1985). Analytical and numerical methods for Volterra equations. *SIAM Studies in Appl. Math. 7,* SIAM, Philadelphia.
58. J. LIOUVILLE (1837). Sur le développement des fonctions ou parties de fonctions en séries dont divers termes sont assujettis à satisfaire à une même équation différentielle du second ordre contenant un paramètre

variable. *J. Math. Pures Appl. (1) 2,* 16-35, 418-436.

59. J.E. LOGAN (1976). *The Approximate Solution of Volterra Integral Equations of the Second Kind,* Ph.D. Thesis, University of Iowa, Iowa City.
60. CH. LUBICH (1983). On the stability of linear multistep methods for Volterra convolution equations. *IMA J. Numer. Anal. 3,* 439-465.
61. CH. LUBICH (1983). Runge-Kutta theory for Volterra and Abel integral equations of the second kind. *Math. Comp. 41,* 87-102.
62. CH. LUBICH (1983). *Fractional Linear Multistep Methods for Abel-Volterra Integral Equations of the Second Kind,* Preprint Nr. 233, Sonderforschungsbereich 123, Universität Heidelberg.
63. D.F. MAYERS (1962). Equations of Volterra type. L. FOX (ed.). *Numerical Solution of Ordinary and Partial Differential Equations,* 165-173, Pergamon Press, Oxford.
64. S. MCKEE (1982). Generalised discrete Gronwall lemmas. *Z. Angew. Math. Mech. 62,* 429-434.
65. SH.E. MIKELADZE (1935). De la résolution numériques des équations intégrales. *Bull. Acad. Sci. URSS VII, NR. 2,* 255-300 (Russian).
66. SH.E. MIKELADZE (1951). On approximate solution of integral equations of volterra type. *Akad. Nauk Gruzin. SSR Trudy Mat. Inst. Razmadze 18,* 315-325 (Russian).
67. W.E. MILNE (1969). *Numerical Solution of Differential Equations* (2nd ed.), Dover, New York.
68. A.F. MONNA (1973). *Functional Analysis in Historical Perspective,* Oosthoek Publ. Co., Utrecht.
69. B. NOBLE (1964). The numerical solution of nonlinear integral equations and related topics. P.M. ANSELONE (ed.). *Nonlinear Integral Equation,* 215-318, University of Wisconsin Press, Madison.
70. B. NOBLE (1971). *A Bibliography on:'Methods for Solving Integral Equations',* MRC Tech. Summ. Reports 1176 (Author Listing), 1177 (Subject Listing), University of Wisconsin, Madison.
71. E.J. NYSTRÖM (1928). Ü'ber die praktische Auflösung van linearen Integralgleichungen und Anwendungen auf Randwertaufgaben der Potentialtheorie. *Soc. Sci. Fenn. Comment. Phys.-Math. 4, No. 15,* 1-52.
72. E. PICARD (1890). Mémoire sur la théorie des équations aux derivées partielles et la méthode des approximations successives. *J. Math. Pures Appl. (4) 6,* 145-210 (cf. 197-200).
73. D. POISSON (1826). Mémoire sur la théorie du magnetisme en mouvement. Oeuvres, III, no. 5, 41-72.
74. P. POUZET (1963). Etude en vue de leur traitement numérique des équations intégrales de type Volterra. *Rev. Française Trait. Inform. Chiffres 6,* 79-112.
75. G. PRASAD (1924). On the numerical solution of integral equations. *Proc. Edinburgh Math. Soc. 42,* 46-59.
76. *Procédures ALGOL en Analyse Numérique I* (1967). Editions Centre National de la Recherche Scientifique, Paris.
77. H.J.J. TE RIELE (1979). Introduction and global survey of numerical

methods for integral equations. H.J.J. TE RIELE (ed.) *Colloquium Numerical Treatment of Integral Equations,* 3-25, Mathematisch Centrum, Amsterdam.

78. H.J.J. TE RIELE (1982). Collocation methods for weakly singular second-kind Volterra integral equations with nonsmooth solution. *IMA J. Numer. Anal. 2,* 437-449.
79. M. SCHLICHTE (1984). *Anwendung eines impliziten Runge-Kutta Verfahrens auf Volterrasche Integralgleichungen zweiter Art mit Faltungskern,* Diplomarbeit, Institut für Angewandte Mathematik, Universität Heidelberg.
80. P.-M. SCHOEDON (1970). *Beiträge zur numerischen Lösung nichtlinearer Volterrascher Integralgleichungen und Integrodifferentialgleichungen mit dem Runge-Kutta Verfahren,* Dissertation, Rheinisch-Westfälische Technische Hochschule Aachen.
81. J.A. SCOTT (1984). *A Unified Analysis of Discretization Methods,* D. Phil. Thesis, University of Oxford.
82. L.F. SHAMPINE (1985). Solving Volterra integral equations with ODE codes. Preprint (to appear in *IMA J. Numer. Anal.*).
83. D. SPOHN (1965). Sur les formules à pas liés dans l'intégration numérique des équations intégrales du type de Volterra. *Quatriéme Congr. de Calcul et de Traitement de l'Informatique AFIRO,* 349-356, Dunod, Paris.
84. Y. SUYAMA, K. NAKAMORI (1952). On numerical solution of the integral equation of Volterra type. *Mem. Fac. Sci. Kyushu Univ. Ser. A 6,* 121-129 (Esperanto).
85. *Symposium on the Numerical Treatment of Ordinary Differential Equations, Integral and Integro-Differential Equations,* Rome (1960), Birkhäuser Verlag, Basel.
86. M. TANFULLA, G. RIBIGHINI (1981). Procedure di stima dell'errore nella risoluzione di equazioni integrali di tipo Volterra. *Riv. Mat. Univ. Parma (4) 7,* 473-487.
87. C.P. TSOKOS, W.J. PADGETT (1974). *Random Integral Equations with Application to Life Sciences und Engineering,* Academic Press, New York.
88. A.F. VERLAN, V.S. SIZIKOV (1978). *Methods for the Solution of Integral Equations Using Computer Programs,* Izdat. 'Naukova Dumka', Kiev (Russian).
89. V. VOLTERRA (1896). Sulla inversione digli integrali definiti. Nota I, II, III, IV. *Opere Matematiche, II,* 216-254, Academia Nazionale dei Lincei, Roma.
90. V. VOLTERRA (1913). *Leçons sur les équations intégrales et les équations intégro-differentielles,* Gauthier-Villars, Paris (cf. 40-45).
91. C. WAGNER (1954). On the numerical solution of Volterra integral equations. *J. Math. Phys. 32,* 289-301.
92. E.T. WHITTAKER (1918). On the numerical solution of integral equations. *Proc. Roy. Soc. London (A) 94,* 367-383.
93. H.M. WILLIAMS, S. MCKEE (1985). Variable stepsize predictor-corrector schemes for second kind Volterra integral equations. *Math. Comp. 44,*

391-404.
94. P.H.M. WOLKENFELT (1982). The construction of reducible quadrature rules for Volterra integral and integro-differential equations. *IMA J. Numer. Anal. 2,* 131-152.
95. P.H.M. WOLKENFELT (1983). Modified multilag methods for Volterra functional equations. *Math. Comp. 40,* 401-316.
96. A. YOUNG (1954). The application of product integration to the numerical solution of integral equations. *Proc. Roy. Soc. London (A) 224,* 561-573.

VITO VOLTERRA

The Amoeba Distributed Operating System (Part 1)

Sape J. Mullender
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

The Amoeba Project is a distributed project on distributed operating systems. The project, which started as the author's PhD research project in 1978 [8], is now a joint project of CWI and Vrije Universiteit in Amsterdam. About a dozen people are working on the project, led by prof. dr. A. S. Tanenbaum (VU) and the author (CWI). This article describes the interprocess communication facilities and protection mechanisms of the Amoeba system. Part 2, which will appear in the next Newsletter, will be devoted to the services provided by the Amoeba Distributed Operating System.

1. Introduction

Distributed information processing has long been practised by living organisms. The human brain, one of the most complicated living organs, functions in a highly distributed manner; different parts of the brain have specialised to perform different functions, such as speech and vision. Yet there does not seem to be any central control in the brain, 'consciousness' cannot be pinpointed to one specific group of brain cells. Not a single function of the brain seems to be impaired when any cell in the brain dies. The individual cells in living organisms die, are replaced by others, and yet, the organism as a whole continues to function uninterruptedly. Most life forms use distributed control of some form or another. Even simple life forms, such as the one-celled *amoebæ*—which have no single 'command centre' to decide where to go and how to get there—are somehow capable of co-ordinated action.

Imitating nature in all aspects, man has finally begun to incorporate the principles of distributed information processing in his most complicated artifacts, computers. In their desire to construct better, faster and more reliable information processing systems, researchers are building networks of many computers which co-operate to preform their task more quickly and more reliably.

The technology for connecting computers is available; many varieties of local-area networks are on the market, and most are fast and reliable.

However, the infrastructure which is necessary to manage and control distributed information has hardly been developed. The subject of our research at CWI is the design of such an infrastructure, a model that allows people to understand distributed computer systems and describe their actions.

Distributed computing is a new research area, one that introduces a whole range of new problems to be solved, problems of managing information systems without global and up-to-date information of their state, of finding ways to prevent inconsistencies in large bodies of data caused by unsynchronised simultaneous changes. Mechanisms must be found for protecting information against unauthorised access. The potential in distributed systems of much greater reliability must be used by designing services that can survive failures of individual components of the system. For some of these problems, solutions had already been found in traditional, centralised operating systems; other problems did not even exist before the advent of distributed computing.

Take the exploitation of parallelism, for instance: If there are two different programs to be run, two processors are evidently more powerful than one; the work can be divided. But this is not so evident if there is only one program to be run. It is then much harder to put the available parallelism to use. Traditional system design methods and software engineering principles do not provide adequate methods of splitting up algorithms in independent parts which can be executed in parallel. Building distributed systems is easy. Using them is hard.

Potentially, systems built up of many processors are more reliable than traditional computers with a single CPU. If the single processor of a centralised system fails, the system comes to a halt. In a distributed system, this does not have to be the case. Every single component of the system can be replicated, so that, no matter what component fails, a subsystem is left behind that can be made to work. If one processing element fails, others can take over the work. If a disk fails, a copy of the information can still be available on another disk.

As it turns out, designing software that exploits this fault-tolerant property of such a configuration is surprisingly difficult. Standard techniques for software development are all based on the assumption that the underlying hardware is infallible. This is a perfectly proper assumption in traditional systems, where, if part of the system fails, the whole system stops working, but it is no longer true in a distributed system.

Distributed systems research concentrates on the problem of structuring the hardware and designing the operating system software in such a way that we can profit from the architecture's two most important potentials, parallelism and fault tolerance.

Distributed control plays a central role in 'avoiding single points of failure.' Specialisation and control cannot be obtained through a simple hierarchical structure as exemplified by most armies. Again, the analogy with nature teaches us that extensive hierarchical systems can exist with a control structure that provides enough redundancy to survive 'simple' failures.

The realisation of distributed control in all parts of the system is a key goal of the research: Any centralised part will be a potential bottleneck when the system grows, and a liability in the face of crashes. It is because of the importance of distributed control that we have named the distributed operating system emerging from our research ***Amoeba***, after that one-celled creature using distributed control to move about.

2. The Amoeba Distributed Operating System

The price of processors and memory is decreasing at an incredible rate. Extrapolating from the current trend, it is likely that a single board containing a powerful CPU, a substantial fraction of a megabyte of memory, and a fast network interface will be available for a manufacturing cost of a few hundred dollars in 1990. Our intention, therefore, has been to do research on the architecture and software of machines built up of a large number of such modules.

In particular, we envision three classes of machines: (1) personal computers consisting of a high-quality bit-mapped display and a few processor-memory modules; (2) departmental machines consisting of hundreds of such modules; and (3) large mainframes consisting of thousands of them. The primary difference between these machines is the number of modules, rather than the type of the modules. In principle, any of these machines can be gracefully increased in size to improve performance by adding new modules or decreased in size to allow removal and repair of defective modules. The software running on the various machines should be in essence identical. Furthermore, it should be possible to connect different machines together to form even larger machines and to partition existing machines into disjoint pieces when necessary, all in a way transparent to the user level software.

This model is superior to the oft-proposed 'Personal Computer Model' (as exemplified by Xerox Parc [5]) in a number of ways. In the personal computer model, each user has a dedicated minicomputer, complete with disks, in the office, or at home. Unfortunately, when people work together on large projects, having numerous local file systems can lead to multiple, inconsistent copies of many files. Also, the noise generated by disks in every office, and the maintenance problems generated by having machines spread all over many buildings can be annoying.

Furthermore, computer usage is very 'bursty': most of the time the user does not need any computing power, but once in a while he may need a very large amount of computing power for a short time (*e.g.*, when recompiling a program consisting of 100 files after changing a basic shared declaration). The fifth-generation computer we propose is especially well suited to bursty computation. When a user has a heavy computation to do, an appropriate number of processor-memory modules are temporarily assigned to him. When the computation is completed, they are returned to the idle pool for use by other users. This contrasts with the Cambridge Distributed Operating System [9], which also has a 'processor bank,' but assigns a processor to a user for the duration of a login session.

A machine of the type described above requires radically different system software than existing machines. Not only must the operating system effectively use and manage a very large number of processors, but the communication and protection aspects are very different from those of existing systems.

Traditional networks and distributed systems are based on the concept of two processes or processors communicating via connections. The connections are typically managed by a hierarchy of complex protocols, usually leading to complex software and extreme inefficiency. (An effective transfer rate of 0.5 megabit/sec over a 10 megabit/sec local network, which is only 5% utilisation, is frequently barely achievable.)

We reject this traditional approach of viewing a distributed system as a collection of discrete processes communicating via multilayer (*e.g.*, ISO) protocols, not only because it is inefficient, but because it puts too much emphasis on specific processes, and by inference, on processors. Instead we propose to base the software design on a different conceptual model—the object model. In this model, the system deals with abstract objects, each of which has some set of abstract operations that can be performed on it.

Associated with each object are one or more *'capabilities'* [3] which are used to control access to the object, both in terms of who may use the object and what operations he may perform on it. At the user level, the basic system primitive is performing an operation on an object, rather than such things as establishing connections, sending and receiving messages, and closing connections. For example, a typical object is the file, with operations to read and write portions of it.

The object model is well-known in the programming languages community under the name of *'abstract data type'* [6]. This model is especially well-suited to a distributed system, because in many cases an abstract data type can be implemented on one of the processor-memory modules described above. When a user process executes one of the visible functions in an abstract data type, the system arranges for the necessary underlying message transport from the user's machine to that of the abstract data type and back. The header of the message can specify which operation is to be performed on which object. This arrangement gives a very clear separation between users and objects, and makes it impossible for a user to directly inspect the representation of an abstract data type by bypassing the functional interface.

A major advantage of the object or abstract data type model is that the semantics are inherently location independent. The concept of performing an operation on an object does not require the user to be aware of where objects are located or how the communication is actually implemented. This property gives the system the possibility of moving objects around to position them close to where they are frequently used. Furthermore, the issue of how many processes are involved in carrying out an operation, and where they are located is also hidden from the user.

It is convenient to *implement* the object model in terms of clients (users) who send messages to services [2, 9, 1]. A service is defined by a set of commands and responses. Each service is handled by one or more server processes that accept messages from clients, carry out the required work, and send back replies. The design of these servers and the design of the protocols they use form an important part of the system software of our proposed fifth-generation computers.

As an example of the problems that must be solved, consider a file server. Among other design issues that must be dealt with are how and where information is stored, how and when it is moved, how it is backed up, how concurrent reads and writes are controlled, how local caches are maintained, how information is named, and how accounting and protection are accomplished. Furthermore, the internal structure of the service must be designed: how many server processes are there, where are they located, how and when do they communicate, what happens when one of them fails, how is a server process organised internally for both reliability and high performance, and so on. Analogous questions arise for all the other servers that comprise the basic system software.

3. Communication Primitives and Protocols

In the literature about computer networks, one finds much discussion of the ISO Reference Model for Open Systems Interconnection (OSI) [12] these days. It is our belief that the price that must be paid in terms of complexity and performance in order to achieve an 'open' system in the ISO sense is much too high, so we have developed a much simpler set of communication primitives, which we will now describe.

Instead of a 7-layer protocol, we effectively have a 4-layer protocol. The bottom layer is the Physical Layer, and deals with the electrical, mechanical and similar aspects of the network hardware. The next layer is the Port Layer, and deals with the location of services, the transport of (32K byte) datagrams (packets whose delivery is not guaranteed) from source to destination and enforces the protection mechanism, which will be discussed in the next section. On top of this we have a layer that deals with the reliable transport of bounded length (32K byte) requests and replies between client and server. We have called this layer the Transaction Layer. The final layer has to do with the semantics of the requests and replies, for example, given that one can talk to the file server, what commands does it understand. The bottom three layers (Physical, Port, and Transaction) are implemented by the kernel and hardware; only the Transaction Layer interface is visible to users. User programs execute in the fourth layer, the Application Layer.

The main function of the Transaction Layer is to provide an end-to-end *message* service built on top of the underlying *datagram* service, the main difference being that the former uses timers and acknowledgements to guarantee delivery whereas the latter does not.

The Transaction Layer protocol is straightforward. A *server process* makes a call to *getreq* (an abbreviation of *get request*) to tell the Transaction Layer it is ready to receive a request from a client. The client sends a request by calling *trans* (for *transaction*), which makes the Transaction Layer send a request and wait until a reply comes back from the server. The client is blocked until this reply arrives. The server, after carrying out the request, returns a reply by a call to *putrep*.

When the client does a *trans*, a packet, or sequence of packets, containing the request is sent to the server, the client is blocked, and a timer is started (inside the Transaction Layer). If the server does not acknowledge receipt of the request packet before the timer expires (usually by sending the reply, but in some special cases by sending a separate acknowledgement packet), the Transaction Layer retransmits the packet again and restarts the timer. When the reply finally comes in, the client sends back an acknowledgement (possibly piggybacked onto the next request packet) to allow the server to release any resources, such as buffers, that were acquired for this transaction. Under normal circumstances, reading a long file, for example, consists of the sequence

From client:	request for block 0
From server:	here is block 0
From client:	acknowledgement for block 0 and request for block 1
From server:	here is block 1
	etc.

The protocol can handle the situation of a server crashing and being rebooted quite easily since each request contains the identity of the file to be read and the position in the file to start reading. Between requests, the server has no 'activation record' or other table entry whose loss during a crash causes the server to forget which files were open, *etc.*, because no concept of an open file or a current position in a file exists on the server's side. Each new request is completely self-contained. Of course for efficiency reasons, a server may keep a cache of frequently accessed *i-nodes*, file blocks *etc.*, but these are not essential and their loss during a crash will merely slow the server down slightly while they are being dynamically refreshed after a reboot.

The Port Layer is responsible for the speedy transmission of 32K byte datagrams. The Port Layer need only do this reasonably reliably, and does not have to make an effort to guarantee the correct delivery of every datagram. This is the responsibility of the Transaction Layer. Our results show that, compared to other approaches, our's leads to significantly higher transmission speeds, due to simpler protocols.

Theoretically, very high speeds are achievable in modern local-area networks. A typical transfer rate of a modern local-area network is 500,000 bytes/sec point-to-point if there were no protocol overhead. In practice, however, speeds of 100,000 bytes per second between user processes have rarely been achieved. Obviously, to achieve higher transmission rates, the overhead of the protocol must be kept very low indeed. To do this, a large datagram size was chosen for the Port Layer, which has to split up the datagrams into small packets that the network hardware can cope with. This approach allows the implementor of the Port Layer to exploit the possibilities that the hardware has to offer to obtain an efficient stream of packets.

Two versions of the algorithm have now been implemented. The one described above has been implemented on the ***Amoeba*** distributed operating system, and achieves over 300,000 bytes a second from user process to user process (using M68000s and a Pronet* ring). A second implementation runs under UNIX,† using 2K byte datagrams, which gets 90,000 bytes/sec across the network between two VAX-750s running a normal load of work, without causing a significant load on the system itself.

4. PORTS

Every service has one or more *ports* [7] to which client processes can send messages to contact the service. Ports consist of large numbers, typically 48 bits, which are known only to the server processes that comprise the service, and to the service's clients. For a public service, such as the system file service, the port will be generally made known to all users. The ports used by an ordinary user process will, in general, be kept secret. Knowledge of a port is taken by the system as *prima facie* evidence that the sender has a right to communicate with the service. Of course the service is not required to carry out work for clients just because they know the port, for example, the public file service may refuse to read or write files for clients lacking account numbers, appropriate authorisation, *etc.*

Communication using ports basically works as follows. One, or several server processes make a call to

getreq(*serverport*, · · ·);

a client, wishing to have some service rendered, calls

trans(*serverport*, · · ·);

The client's Transaction Layer generates a (unique) *reply port* for the client,

* PRONET is a trademark of Proteon Associates, Inc.
† UNIX is a Trademark of AT&T Bell Laboratories.

finds an active server process on *serverport* and sends a message containing

{*serverport*, *replyport*, · · · }

to the server. The server processes the request, and returns a reply in a message

{*replyport*, *serverport*, · · · }

The two ports provide a unique identification of the transaction.

The difference between this approach and that taken by conventional interprocess communication protocols is that, in principle, messages are addressed to a service name or *port*, not to a machine, or a process. Obviously, in Amoeba, clients never need to know where a service is implemented, or how many server processes there are. This is none of their business; it is part of the *implementation* of the 'abstract data type' that is the service.

The transaction mechanisms, however, must deliver requests and replies to specific processes on specific machines. Obviously, inside the Port Layer—which is, after all, responsible for message delivery—ports must be mapped onto networks addresses. So, when a client calls *trans*, the client's Port Layer must *find* a network node where a *getreq* on the matching port is outstanding.

In the local-area network, the technique for *locating* a port is the following. The client broadcasts a tiny message, saying '*anyone listening on port* x*?*', to which servers listening on that port reply '*port* x *is at machine* y*!*' A similar technique is used for locating the client for the reply message.

Locating ports is inefficient. It can be sped up, however, by a simple technique which finds many useful applications in distributed systems: the *hint*. In this particular case, hints are stored in a little table of (*port*, *network address*) pairs in every host and they say in effect: '*If you're looking for port* x, *why don't you try network address* y*?*' If the hint works, a port has been located without sending any extra messages; if not, a message is returned saying that the port is not known at that address. In the latter case, the hint is scratched out, and a broadcast locate is done. Incoming packets contain a source address, so they provide a free hint for their source port.

Although the port mechanism provides a convenient way to provide partial authentication of clients ('if you know the port, you may at least talk to the service'), it does not deal with the authentication of servers. The primitive operations offered by the system are *trans*, *putreq* and *getrep*. Since everyone knows the port of the file server, as an example, how does one ensure that malicious users do not execute *getreq*s on the file server's port, in effect impersonating the file server to the rest of the system?

One approach is to have all ports manipulated by kernels that are presumed trustworthy and are supposed to know who may *getreq* from which port [2, 11].

We reject this strategy because some machines, *e.g.*, personal computers connected to larger multimodule systems, may not be trustworthy, and also because we believe that by making the kernel as small as possible, we can enhance the reliability of the system as a whole. Instead, we have chosen a different solution that can be implemented in either hardware or software.

In the hardware solution, we need to place a small interface box, which we call an F-box (Function-box) between each processor module and the network. The most logical place to put it is on the VLSI chip that is used to interface to the network. Alternatively, it can be put on a small printed circuit board inside the wall socket through which personal computers attach to the network. In those cases where the processors have user mode and kernel mode and a trusted operating system running in kernel mode, it can also be put into operating system software. In any event, we assume that somehow or other all packets entering and leaving every processor undergo a simple transformation that users cannot bypass.

The transformation works like this. Each port is really a pair of ports, P, and G, related by: $P = F(G)$, where F is a (publicly-known) one-way function [13, 10, 4] performed by the F-box. The one-way function has the property that given G it is a straightforward computation to find P, but that given P, finding G is so difficult that the only approach is to try every possible G to see which one produces P. If P and G contain sufficient bits, this approach can be made to take millions of years on the world's largest supercomputer, thus making it effectively impossible to find G given only P. Note that a one-way function differs from a cryptographic transformation in that the latter must have an inverse to be useful, but the former has been carefully chosen so that no inverse can be found.

Using the one-way F-box, the server authentication can be handled in a simple way, illustrated in FIGURE 1. Each server chooses a get-port, G, and computes the corresponding put-port, P. The get-port is kept secret; the put-port is distributed to potential clients, or, in the case of public servers, is published. When the server is ready to accept client requests, it does a *getreq*(G, *cap*, *req*). The F-box then computes $P = F(G)$ and waits for packets containing P to arrive. When one arrives, it is given to the appropriate process. To send a packet to the server, the client merely does *trans*(*cap*, *req*, *rep*). *Cap* is a *capability**, giving the identity of the object the user wants to access, which contains the *port* field, P, of the service managing the object. This will cause a datagram to be sent by the local F-box with P in the destination-port field of the header. The F-box on the sender's side does not perform any transformation on the P field of the outgoing packet.

* Capabilities are explained in the next section.

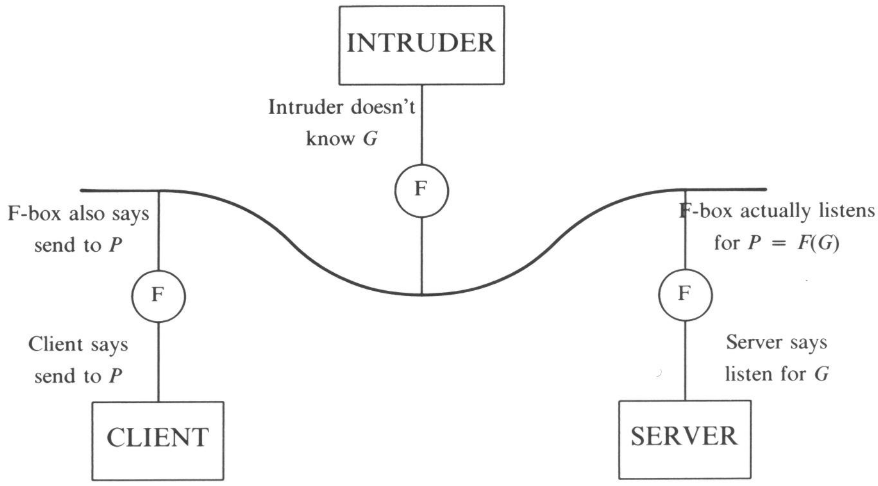


FIGURE 1.

Now let us consider the system from an intruder's point of view. To impersonate a server, the intruder must do *getreq*$(G, \cdots)$. However, G is a well-kept secret, and is never transmitted on the network. Since we have assumed that G cannot be deduced from P (the one-way property of F) and that the intruder cannot circumvent the F-box, he cannot intercept packets not intended for him. Replies from the server to the client are protected the same way: The client's Transaction Layer picks a get-port for the reply, say, G', and the client's F-box transforms G' into $P' = F(G')$ in the request packet for the server to use as the put-port to send the reply to.

The presence of the F-box makes it easy to implement digital signatures for still further authentication, if that is desired. To do so, each client chooses a random signature, S, and publishes $F(S)$. The F-box must be designed to work as follows. Each packet presented to the F-box contains three special header fields: destination (P), reply (G'), and signature (S). The F-box applies the one-way function to the second and third of these, transmitting the three ports as: P, $F(G')$, and $F(S)$, respectively. The first is used by the receiver's F-box to admit only packets for which the corresponding *getreq* has been done, the second is used as the put-port for the reply, and the third can be used to authenticate the sender, since only the true owner of the signature will know what number to put in the third field to ensure that the publicly-known $F(S)$ comes out.

It is important to note that the F-box arrangement merely provides a simple *mechanism* for implementing security and protection, but gives operating system designers considerable latitude for choosing various *policies*. The mechanism is sufficiently flexible and general that it should be possible to put it into hardware without precluding many as-yet-unthought-of operating systems to be designed in the future.

5. Capabilities

In any object-based system, a mechanism is needed to keep track of which processes may access which objects and in what way. The normal way is to associate a capability with each object, with bits in the capability indicating which operations the holder of the capability may perform. In a distributed system this mechanism should itself be distributed, that is, not centralised in a single monolithic 'capability manager'. In our proposed scheme, each object is managed by some service, which is a user (as opposed to kernel) program, and which understands the capabilities for its objects.

SERVER	OBJECT	RIGHTS	CHECK

Figure 2.

A capability typically consists of four fields, as illustrated in Figure 2:

1. The put-port of the service that manages the object
2. An Object Number meaningful only to the service managing the object
3. A Rights Field, which contains a 1 bit for each permitted operation
4. A Check Field for protecting each object

The basic model of how capabilities are used can be illustrated by a simple example: a client wishes to create a file using the file service, write some data into the file, and then give another client permission to read (but not modify) the file just written. To start with, the client sends a message to the file service's put-port specifying that a file is to be created. The request might contain a file name, account number and similar attributes, depending on the exact nature of the file service. The server would then pick a random number, store this number in its object table, and insert it into the newly-formed object capability. The reply would contain this capability for the newly created (empty) file.

To write the file, the client would send a message containing the capability and some data. When the *write* request arrived at the file server process, the server would normally use the object number contained in the capability as as index into its tables to find the object. For a Unix-like file server, the object number would be the i-node number, which could be used to locate the i-node.

Several object protection systems are possible using this framework. In the simplest one, the server merely compares the random number in the file table (put there by the server when the object was created) to the one contained in the capability. If they agree, the capability is assumed to be genuine, and all operations on the file are allowed. This system is easy to implement, but does not distinguish between *read*, *write*, *delete*, and other operations that may be performed on objects.

However, it can easily be modified to provide that distinction. In the modified version, when a file (object) is created, the check field is computed by applying a one-way function to Object Number, Rights Field, and the Random Number stored with the object. When the capability is returned for use, the server uses the object number to find the file table and hence the random number. If the result of recomputing the Check Field leads to the Check Field in the capability, it is almost assuredly valid, and the Rights Field can be believed. Clearly, an encryption function that mixes the bits thoroughly is required to ensure that tampering with the Rights Field also affects the Check Field.

When this modified protection system is used, the owner of the object can easily give an exact copy of the capability to another process by just sending it the bit pattern, but passing, say, read-only access, is harder. To accomplish this task, the process must send the capability back to the server with a bitmap saying which bits to strip off the Rights Field. By choosing the bit mask carefully, the capability owner can mask out any operations that the recipient is not permitted to carry out.

REFERENCES

1. J. E. BALL, E. J. BURKE, I. GERTNER, K. A. LANTZ, AND R. F. RASHID, (1979). Perspectives on Message-Based Distributed Computing, *Proc. IEEE.*
2. D. R. CHERITON AND W. ZWAENPOEL, (October 1983). The Distributed V Kernel and its Performance for Diskless Workstations, *Operating Systems Review*, 17.5, 129-140.
3. J. B. DENNIS AND E. C. VAN HORN, (March 1966). Programming Semantics for Multiprogrammed Computations, *Comm. ACM*, 9.3, 143-155.
4. A. EVANS, W. KANTROWITZ, AND E. WEISS, (August 1974). A User Authentication Scheme Not Requiring Secrecy in the Computer, *Comm. ACM*, 17.8, 437-442.
5. B. W. LAMPSON AND R. F. SPROULL, (1979). An Open Operating System For A Single User Machine, *Proc. Seventh Symp. on Oper. Syst. Prin.*, 98-105.
6. B. LISKOV AND S. ZILLES, (April 1974). Programming With Abstract Data Types, *SIGPLAN Notices*, 9, 50-59.
7. S. J. MULLENDER AND A. S. TANENBAUM, (1984). Protection and

Resource Control in Distributed Operating Systems, *Computer Networks*, 8.5,6, 421-432.

8. S. J. MULLENDER, (October 1985). *Principles of Distributed Operating System Design*: SMC, Amsterdam.
9. R. M. NEEDHAM AND A. J. HERBERT, (1982). *The Cambridge Distributed Computer System*: Addison-Wesley, Reading, Ma..
10. G. B. PURDY, (August 1974). A High Security Log-in Procedure, *Comm. ACM*, 17.8, 442-445.
11. R. RASHID (May 1981). Accent: A Network Operating System for SPICE/DSN, *Tech. Rept., Computer Science Dept.*, Carnegie-Mellon University.
12. A.S. TANENBAUM (1981). The ISO-OSI reference model, *IR-71*, Vrije Universiteit, Amsterdam.
13. M. V. WILKES, (1968). *Time-Sharing Computer Systems*: American Elsevier, New York.

B for the IBM PC

A prototype version of *B* (see Newsletter 3) is now available for the IBM PC and compatibles. This version is functionally equivalent to the one currently distributed for UNIX systems.

FEATURES OF THE IMPLEMENTATION

- The full *B* language as described in the Draft Proposal is implemented.
- The structured editor is used to enter and edit units, immediate commands, input to READ commands, and permanent targets.
- The editor suggests possible command continuations and closing brackets. It uses function and arrow keys to move the focus around, change its size, etc.. You can *undo* the last 20 key strokes. Text can be moved or duplicated within or between units and immediate commands. A sequence of keystrokes can be recorded, and played back later. You can recall the last command. (A more detailed description of the *B* editor can be found in the first *B Newsletter*.) By default all editing operations are bound to single keys. You can rebind the editing operations to other than the default keys, to suit your own taste, or to overcome deficiencies in your particular keyboard.
- Since different compatibles have different ways of addressing the screen, and not all screens have the same size, you can define an environment variable to reconfigure *B* for use with the 'ANSI' screen driver, or for different screen widths and heights. There is a program supplied to help you decide what sort of screen you have.
- There are utility programs for such things as workspace recovery, and for listing the units in a workspace.

Small version

The full implementation needs at least 384 K bytes. There is a smaller version available, without the built-in editor, for those with only 256 K bytes. In this case commands must be typed in in full. To edit units the EDLIN editor is used.

Documentation

The documentation sent with the package includes 'The *B* Programmer's Handbook' and a quick reference guide. The book describes the *B* language proper, the use of the system, the editor commands, and the use of the other utilities.

Required system configuration

In order to run the *B* system you must have an IBM PC or compatible with

- at least 384 K bytes, or 256 K bytes for the small version (this includes space used for MS-DOS);
- MS-DOS version 2.x (we've only tested it with 2.0 and 2.11, but it should also run on MS-DOS 3.x or higher);
- one double-sided disk drive.

The system is expected to run on most IBM-PC compatible computers, and we have tried it on many, but we cannot guarantee this because we haven't tried them all. We know, however, that the system runs at least on the following compatibles: Olivetti M24, Apricot Portable, Apricot F1, GRID Case, Tulip Extend, Goupil.

(If your copy runs fine on a machine which is not on the list, please tell us so we know we can extend the list. If your copy doesn't run, also tell us and we'll try to see what is the cause.)

Beware

IT IS NOT A PRODUCTION VERSION!

The system is sometimes slow, and imposes severe limits on the maximum sizes of the targets and units in the work space. We *do* appreciate reports of bugs, but we don't promise we'll fix them: we'll do what we can.

Finally

The disk is not copy protected. You may make copies, and give them away, as long as you don't sell the copies, and as long as these same conditions are passed on to the people you give copies to. Fair enough?

HOW TO ORDER *B* for the IBM PC:

To order the prototype of the implementation of *B* for the IBM PC or compatibles, running under MS-DOS versions 2.0 (or higher), you should fill in the order form below, and send it to:

B Group, PC distribution
Informatics / AA
CWI
POB 4079
1009 AB Amsterdam
The Netherlands

To cover materials and handling, you should either enclose a cheque or money order, payable to Stichting Mathematisch Centrum - Amsterdam, or (if you live in The Netherlands) transfer to the postgiro account below.

You will then receive:

- a floppy with the binary;
- The *B* Programmer's Handbook;
- a *B* Quick Reference Card.

ORDER FORM

Please send me a copy of the prototype *B* system for the IBM PC, including documentation.

☐ I enclose a cheque or international money order, payable to Stichting Mathematisch Centrum - Amsterdam, for Dfl 100 (or US $ 35).

☐ I have transferred Dfl 100 to postgiro account 462890, Stichting Mathematisch Centrum - Amsterdam, mentioning "*B* voor de IBM PC".

Name: ....

Firm/Institute: ....

Address: ....

....

Country: ....

Telephone: ....

Network address: ....

Machine(s): ☐ IBM PC ☐ IBM XT ☐ Olivetti M24
☐ Apricot Portable ☐ Apricot F1 ☐ other: ......

Required media:

☐ 5¼" double sided, double density floppy disk
☐ 3½" double sided floppy disk

Required version:

☐ full implementation (at least 384 K bytes)
☐ small version (only 256 K)

Signature and Date:

Abstracts

of Recent CWI Publications

When ordering any of the publications listed below please use the order form at the back of this issue.

CWI Tract 22. J.C.S.P. van der Woude. *Topological Dynamix.*
AMS 54H20; 298 pp.

Abstract: In this book several topics from abstract topological dynamics are dealt with. The (mostly implicit) main theme is the structure theory for minimal flows, following the tradition set by the works of (among others): J. Auslander, R. Ellis, H. Furstenberg, S. Glasner and W.A. Veech. The central notions are: quasifactors of minimal flows; (weak) disjointness of flow homomorphisms; the equicontinuous structure relation. For notation and consistent reference an extensive introduction to the basics of abstract topological dynamics is given in Chapter I. Chapter II deals with the induced action on the hyperspace of the phase space. $\mathcal{F}$-topology techniques and the equicontinuous structure relation are studied in Chapter III. In Chapters IV and V highly proximal homomorphisms are studied and related to Gleason extensions and orbit closures of certain closed subsets of the universal minimal flow. Disjointness and disjointness relations are the main subject of Chapter VI, while Chapter VII deals with weak disjointness related to invariant measures and the equicontinuous structure relation. The final Chapter deals with a strong form of regional proximality, connecting prolongational limit sets with the equicontinuous structure relation. The underlying problem is the transitivity of the regionally proximal relation.

CWI Tract 23. A.F. Monna. *Methods, Concepts and Ideas in Mathematics: Aspects of an Evolution.*
AMS 01-02. 170 pp.

Abstract: This book consists of three parts which are basically a revised version - in English - of three papers published earlier in French by the same author in the series: 'Communication of the Mathematical Institute Rijksuniversiteit Utrecht', scil.: 'L'algébrisation de la Mathématique' (1977), 'Evolutions de problèmes d'existence en Analyse' (1979) and 'Evolutions en Mathématique' (1981). The first part, entitled: 'The algebraization of Mathematics' deals with the penetration of algebra in the different regions of Mathematics'; 'Algebra' considered as 'the study of explicit structure of postulationally defined systems closed with respect to one or more operations'. The author illustrates this phenomenon with the help of many examples from different regions of mathematics: geometry, topology, integration theory, Lie groups, number theory etc. He also discusses the role of the limit concept in this context and the formal aspects of the derivative. The second part: 'Evolutions of existence problems' broaches the problem concerning existence, especially in analysis. The author sketches the chaning character of analysis since the last decades of the past century: 'classical' analysis had a clearly constructive character, whereas in 'modern' analysis non-constructive methods and existence theorems prevail. As a starting point the author considers Cantor's set theory. Again many examples elucidate the author's standpoint, e.g. an extensive description of the 'Collection Borel', a characterization of the 'Polish school' and the 'Infinitär Kalkül of du Bois-Reymond', to give only an impression. Where the author gives his own opinion, he does this with great reserve. The third part: 'The evolution of Mathematics' falls apart into two sections. In the first one, the author continues his reflections on existence and distinguished 'strong' existence from 'weak' existence, that means a distinction between constructive and non-constructive existence proofs. Much attention is devoted to the origin and the development of fundamental notions such as: construction, axiom, function, group etc. In this context the author considers the role of geometry. In his opinion geometrical insight and geometrical presentation paved the way to abstract theories and generalizations just as earlier physical insight showed the way. At the end of each of the three parts there is a number of notes referring to the related passages in the text. An extensive bibliography (160 titles) and an index of mathematicians conclude the book.

CWI Tract 24. J.C.M. Baeten. *Filters and Ultrafilters over Definable Subsets of Admissible Ordinals.*

AMS 03D60, 03E45, 03E55, 04A20; 85 pp.

Abstract: The search for a recursive analogue of a measurable cardinal leads to a study of ordinals that have a filter, which is complete, normal or an ultrafilter on a Boolean algebra of definable subsets, not on the whole power set. The study of these so-called definable filters combines techniques from definability theory, set theory and recursion theory, and uses the hierarchy of constructible sets. The existence of definable filters is related to admissibility, and we find that the existence of a definable normal (ultra)filter is not equivalent to the existence of a definable (ultra)filter. We look at the analogues of certain classical filters, namely the co-finite filter and the normal filter of closed unbounded sets. We prove that on a countable ordinal, we can extend a definable filter to a definable ultrafilter, and a definable normal filter to a definable normal ultrafilter.

CWI Tract 25. A.W.J. Kolen. *Tree Network and Planar Rectilinear Location Theory.*

AMS 90C10, 05C05, 05C70; 85 pp.

Abstract: Many location problems on networks can be solved as one or as a sequence of covering problems. It is shown that chordal graphs and totally-balanced matrices are useful tools in finding strong duality results and polynomial time algorithms for tree network location problems. For planar location problems using the rectilinear (Manhattan) distance it is shown that Farkas' lemma can be used successfully.

CWI Tract 27. A.J.M. van Engelen. *Homogeneous Zero-Dimensional Absolute Borel Sets.*

AMS 54H05, 03E15; 133 pp.

Abstract: Topological characterizations of such well-known homogeneous zero-dimensional spaces (here, space means: separable metric space) as the Cantor set $\mathbb{C}$, the irrationals $\mathbb{P}$, and the rationals $\mathbb{Q}$ have been known since the beginning of this century. E.g. 'if X is a non-empty zero-

dimensional compact space without isolated points, then X is homeomorphic to $\mathbf{C}$'. Much later, around 1980, some other homogeneous zero-dimensional absolute Borel sets were characterized by van Mill and van Douwen. In this monograph, similar characterizations are given of *all homogeneous zero-dimensional absolute Borel sets;* there turn out to be ω_1 topological types of such spaces. The characterization of the Cantor set above consists largely of a precise indication of its place in the Borel hierarchy (compactness); in the same way, our characterizations mainly consist of a description of the level in the Borel hierarchy of the spaces under consideration. However, the 'usual' Borel hierarchy ($F_\sigma, G_\delta, F_{\sigma\delta}, G_{\delta\sigma}$, etc.) is not sufficiently fine to distinguish between the different homogeneous spaces, and therefore we use (existing) refinements of the Borel hierarchy, viz. the hierarchy of small Borel classes (Kuratowski), and the Wadge hierarchy (Wadge). Using techniques for extending homeomorphisms on nowhere dense sets we arrive at *internal* topological characterizations of those absolute Borel sets that are both of class $F_{\sigma\delta}$ and of class $G_{\delta\sigma}$, in terms of the small Borel classes. The Wadge hierarchy, based on reduction by continuous mappings, is used for Borel sets of higher class; here the characterizations, although topological, are not internal. As an application of our results, we prove that non-trivial *rigid* zero-dimensional absolute Borel sets do not exist, answering a question of van Douwen.

CWI Syllabus 10. *Vacation-course 1986: Matrices.*
162 pp. (in Dutch)

Abstract: Matrices appear in almost all parts of mathematics and there are many ways in which they can be used. In this course (for high school mathematics teachers) a number of applications of matrices will be covered. Many different types of matrices (real, positive, complex, integer, partitioned, ...) exist as well as many classes of matrices with special properties, each requiring its own 'theory'. However, in the syllabus, the emphasis is on applications. Some questions arising outside of mathematics are dealt with that can be answered by use of matrices, but without requiring prior knowledge of their theoretical properties.

CS-R8610. T. Budd. *The cleaning person algorithm.*
CR E.2, D.3.3, D.3.2; 12 pp.; **key words:** programming language implementation, data types and structures, B.

Abstract: The language B is intended to provide a powerful tool that can nevertheless be used by novice programmers in the solution of nontrivial problems. Part of the design of B involves removing from the programmer's concern limitations imposed by such factors as machine word size or memory size. This paper describes an algorithm that permits values to migrate easily between primary and secondary memory (or disk), permitting the B system to act as if the amount of memory was essentially limitless.

CS-R8611. E. Kranakis & P.M.B. Vitányi. *Distributed control in computer networks and cross-sections of colored multidimensional bodies.*
AMS 68C05, 68C25, 26B15, 28A75; CR C.2.1, F.2.2, G.2.2; 13 pp.; **key words:** distributed match-making, computer network, distributed control, name server, mutual exclusion, colored body, measure.

Abstract: The number of messages to match a pair of processes in a multiprocessor network with mobile processes is a measure for the cost of setting up temporary communication between processes. We establish lower bounds on the average number of point-to-point transmissions between any pair of nodes in this context. The present analysis allows for the possibility of multiple transmissions (as opposed to a single one) between any two nodes, and also for the possibility of multiple queries (as opposed to the two, i.e. post and a single query, considered before). Applications of the results include lower bounds on the number of messages for distributed s-matching, that is, matching a group of s processes, and distributed s-mutual exclusion, that is, s-1 processes

may enter a critical section simultaneously, but s processes may not, for $s \geq 2$. The idea of the proof of the combinatorial result needed for this analysis is further extended to obtain a lower bound on the average number of colors occurring in random cross-sections of colored, multidimensional bodies in terms of the total (multidimensional) volume of each color in the whole body.

CS-R8612. Ming Li, Luc Longpré & P.M.B. Vitányi. *The power of the queue.*
AMS 68C40, 68C25, 68C05, 94B60, 10-00; CR F.1.1, F.1.3, F.2.3; 17 pp.; **key words:** tape, stack, queue, pushdown stores, determinism, nondeterminism, off-line, time complexity, lower bound, upper bound, simulation algorithmic information theory, Kolmogorov complexity.

Abstract: Queues, stacks (pushdown stores), and tapes are storage models which have direct applications in compiler design and the general design of algorithms. Whereas stacks (pushdown store or last-in-first-out storage) have been thoroughly investigated and are well understood, this is much less the case for queues (first-in-first-out storage). This paper contains a comprehensive study comparing queues to stacks and tapes. We address off-line machines with a one-way input, both deterministic and nondeterministic. The techniques rely on algorithmic information theory (Kolmogorov Complexity).

CS-R8613. P.J.F. Lucas. *Knowledge representation and inference in rule-based systems.*
AMS 69K11, 69K14; CR I.2.1, I.2.4; 17 pp.; **key words:** expert systems, knowledge-based systems, inference.

Abstract: In this paper a review is presented of various approaches to representing and applying human knowledge in expert systems, in particular in rule-based systems. The paper also provides an introduction to some equivalent methods of representation. Some emphasis is put on low-level operations and also on inference procedures that are applied in extracting useful knowledge from a knowledge base. This investigation is partly based on work done in the design and the implementation of the DELFI-2 system at Delft University of Technology and recently at the Centre of Mathematics and Computer Science. It has been particularly influenced by concepts from logic programming.

CS-R8614. J.W. Klop & E. Kranakis. *Lower bounds for a class of Kostka numbers.*
AMS 05A10, 05A20; 9 pp.; **key words:** 0,1 matrix, binomial coefficient, lower bound, upper bound, Stirling's formula, binary entropy, Kostka numbers.

Abstract: A simple proof of a lower bound on the number of $2m \times 2m$ matrices with 0,1 entries and each of whose rows and columns adds to the fixed sum m is presented. In fact, it is shown that for any fixed $0 < \lambda < \frac{1}{2}$ the number of such matrices is asymptotically at least $\binom{2m}{m}^{m + \lambda m}$.
The inductive proof employed in the present paper might also turn out to be useful in obtaining lower bounds for other types of Kostka numbers.

CS-R8615. M.L. Kersten & F.H. Schippers. *A general object-centered database language; a preliminary definition.*
AMS 69D42, 69H23, 69K14; CR D.3.2, H.2.3, I.2.4; 21 pp.; **key words:** object-oriented languages, database management, knowledge representation.

Abstract: This report describes the programming language *Godel,* intended for the construction of knowledge based applications. *Godel* uses both the object-centered, the rule-oriented, and the procedural programming paradigms. These paradigms are used in an unconventional way, thereby

simplifying the maintenance of complex relationships among (static) objects and modelling dynamic behaviour through actor-like objects, called *guardians.* This report is a working document. It focuses on the syntax and it presents an informal semantic definition. In-depth discussions of topics such as the concurrency philosophy and storage techniques used are presented separately. The language definition given here assumes a teletype-like user interface, which simplifies the language specification and its implementation. A functional prototype has been implemented in C-PROLOG under UNIX BSD4.2.

CS-R8616. L.C. van der Gaag. *PROLOG: an expert system building tool.*
AMS 69K11, 69D42; CR I.2.1, D.3.2; 14 pp.; **key words:** expert systems, PROLOG, logic programming.

Abstract: For several years, LISP has been the most popular programming language for artificial intelligence. PROLOG, however, is rapidly becoming the second most popular artificial intelligence language; for several applications, PROLOG is even preferred to LISP. In this paper, the suitability of PROLOG as an expert system building tool is demonstrated: a small expert system shell is discussed, and compared to the DELFI-2 system, after the example of which the PROLOG system has been developed.

CS-R8617. J.A. Bergstra, J. Heering & P. Klint. *Module algebra.*
AMS 68B10; CR D.2.0, D.2.2, D.3.3, F.3.2; 35 pp.; **key words:** algebraic specification, first-order specification, signature, module algebra, module composition, signature expression, module expression, Craig interpolation lemma, information hiding, abstraction, export, union of modules, renaming, visible signature.

Abstract: An axiomatic algebraic calculus of modules is given which is based on the operators *combination/union, export, renaming,* and *taking the visible signature.* Four different models of module algebra are discussed and compared.

CS-R8619. J.A. Bergstra & J.V. Tucker. *Algebraic specifications of computable and semicomputable datatypes.*
AMS 68B10; CR D.2.0, D.2.2, D.3.3, F.3.2; 60 pp.; **key words:** algebraic specification, initial algebra semantics, computable algebra, hidden functions.

Abstract: An extensive survey is given of the properties of various specification mechanisms based on initial algebra semantics.

CS-R8620. J. Heering, J. Sidi & A. Verhoog (eds.). *Generation of interactive programming environments - GIPE.* Intermediate report.
AMS 68B99; CR D.2.1, D.2.6, D.3.1, D.3.4, F.3.2; 500 pp.; **key words:** generation of programming environments, language definition, syntax definition, type checking, inference rule semantics, structured operational semantics, algebraic semantics.

Abstract: The objective of the GIPE-project is to realize a prototype system for generating interactive programming environments from formal language definitions. Partners in this five-years project, which has started in November 1984, are BSO/Automation Technology (Utrecht), CWI (Amsterdam), INRIA (Rocquencourt/-Sophia-Antipolis), and SEMA (Montrouge). In this intermediate report we describe a common development environment, various language definition formalisms, and the environment generator itself.

CS-R8621. H.P. Barendregt, J.R. Kennaway, J.W. Klop & M.R. Sleep. *Needed*

reduction and spine strategies for the lambda calculus.
AMS 03B40, 68Q99; CR D.1.1, F.4.1; 37 pp.; **key words:** lambda calculus, reduction strategy, needed redex, spine redex, strictness analysis.
Abstract: A redex R in a lambda-term M is called *needed* if in every reduction of M to normal form (residual of) R is contracted. Among others the following results are proved. 1) R is needed in M if and only if R is contracted in the leftmost reduction path of M. 2) Let $\Re : M_0 --^{R}0 \rightarrow M_1 --^{R}1 \rightarrow M_2 --^{R}2 \rightarrow .....$ be such that $\forall i \; \exists j \geqslant i \; R_j$ is needed in M_j. Then $\Re$ is normalising, i.e. if M_0 has a normal form, then $\Re$ is finite. 3) Neededness is an undecidable property, but has several efficiently decidable approximations, various versions of the so-called *spine* redexes.

CS-R8622. J.C. Ebergen. *A technique for designing delay-insensitive VLSI circuits.*
AMS 68B10, 68D37, 68FXX, 94C99; CR B.6.1, B.7.1, F.1.1; 12 pp.; **key words:** delay-insensitive circuit, VLSI design, parallelism, trace semantics, specification, decomposition.
Abstract: A technique for the hierarchical design of delay-insensitive circuits is presented. The techniques are developed by means of the trace-theory formalism. The design consists of the formulation of a specification and its decomposition into basic elements. Parallelism is allowed in a specification. The notion of delay-insensitive circuit is formalized. Three examples are given to illustrate the technique.

CS-N8601. N.W.P. van Diepen. *Algebraic specification of a language with goto-statements.*
AMS 68BXX; CR D.2.1, D.3.1, F.3.1, F.3.2; 12 pp.; **key words:** algebraic specifications, initial algebra semantics, goto-statements, modular specifications.
Abstract: The algebraic specification of the semantics of SMALL - a programming language designed to demonstrate specifications in denotational semantics - is given. Focus of attention are the specification of the semantics of goto-statements and the modular build-up of a language specification.

CS-N8602. J.C. van Vliet & J.B. Warmer. *An annotated bibliography on document processing.*
AMS 68K05; CR H.4.1; 12 pp.; **key words:** word processing.
Abstract: This report contains an annotated bibliography on document processing.

CS-N8603. P. Urzyczyn. *Dining philosophers and process algebra.*
AMS 68N10, 68C01, 68D25; 18 pp.; **key words:** process algebra, dining philosophers, liveness, fairness, probabilistic fairness, fair abstraction.
Abstract: We discuss a lifeness property for merges of regular processes, and we apply this property to show correctness of a dining philosophers protocol by means of process algebra.

CS-N8604. S. van Egmond & F.C. Heeman. *Inform: prototype of an interactive formula editor.* (In Dutch)
AMS 68U15; CR I.7.2; 45 pp.; **key words:** text editing, text formatters, typesetting, mathematical formulae, interactive editing.
Abstract: At the CWI, a project is under way concerning interactive document preparation. Aim of

the project is to make a system which allows the user to edit a formatted version of a document on the screen. Documents may contain text, tables, mathematical formulae and pictures. This report describes the design and implementation of an interactive system for editing mathematical formulae.

CS-N8605. L.C. van der Gaag & P.J.F. Lucas. *Introduction to PROLOG.* (In Dutch)
AMS 69K15, 69D42; 29 pp.; **key words:** PROLOG, programming languages.
Abstract: From the fields of both theoretical research and applications, there is a growing interest in the programming language PROLOG. PROLOG is a practical representation of the principles of logic programming; this has resulted in a number of characteristics which make the language highly suitable for symbolic computation. This paper introduces the foundations of PROLOG and its application. Furthermore, some successful applications of PROLOG in a number of areas are discussed.

CS-N8606. A. Eliëns. *Semantics for Occam.*
AMS 68B10, 68C01; CR D.1.3, D.3.1, F.1.2, F.3.2, F.3.3; 63 pp.; **key words:** Occam, transputer, real time, communication, concurrency, transition-systems, operational semantics, denotational semantics, alternation, event-structures.
Abstract: A brief description of the language Occam and its relation to the transputer is given. The problems in specifying a semantics dealing with the real-time instruction *WAIT a period of time* and the possibility of allocating distinct processes to distinct processors are indicated. A variety of semantics is presented, notably a linear time operational semantics on the basis of a transition-system in the style of Plotkin, a branching time denotational semantics in the tradition of De Bakker and Zucker and a metric denotational semantics based on the concept of alternation as put forward by Chandra, Kozan and Stockmeyer. One of the aims of developing the latter semantics was to investigate the possibility of an event-structure like semantics as proposed by Reisig and Winskel in a metric denotational framework as developed by De Bakker and Zucker. A sketch is given of how to interrelate the semantics.

OS-R8604. J.M. Schumacher. *Transformations of linear systems under external equivalence.*
AMS 93B17, 93B20, 93C35, 34A30, 15A22, 15A36; 21 pp.; **key words:** linear systems, system equivalence, state space, minimal representation, feedback connection.
Abstract: We consider systems of linear differential and algebraic equations in which some of the variables are distinguished as 'external variables'. Two systems are called equivalent if the set of solutions for the external variables is the same for both systems. We give an operational form for this definition of equivalence, i.e., we describe a set of system transformations having the property that two systems are equivalent if and only if they can be taken into each other by transformations from that set. Next, an algorithm is described to transform a given system in general form to a system in minimal state space form. This algorithm differs from existing methods in that it first takes the equations to first-order form, so that each subsequent step can be formulated and interpreted in state space terms. We also compute the 'structure indices' in terms of a state space description in non-minimal form and use this to prove the minimality of the end result of the algorithm. Finally, an application is shown to the problem of ill-posedness of feedback connections.

OS-R8605. O.J. Boxma & F.G. Forst. *Minimizing the expected weighted number of tardy jobs in stochastic flow shops.*
AMS 90B35; 10 pp.; **key words:** stochastic sequencing, tardiness, flow shop.

Abstract: This paper is devoted to two types of stochastic scheduling problems, one involving a single machine and the other involving a flow shop consisting of an arbitrary number of machines. In both problem types, all jobs to be processed have due dates, and the objective is to find a job sequence that minimizes the expected weighted number of tardy jobs. For the single-machine case, sufficient optimality conditions for job sequences are derived for various choices of due date and processing time distributions. For the case of a flow shop with an arbitrary number of machines and identically distributed due dates for all jobs, we prove the following intuitively appealing results: (i) when all jobs have the same processing time distributions, the expected weighted number of tardy jobs is minimized by sequencing the jobs in decreasing order of the weights; (ii) when all weights are equal, the jobs should be sequenced according to an increasing stochastic ordering of the processing time distributions.

OS-R8606. O.J. Boxma & W.P. Groenendijk. *Pseudo-conservation laws in cyclic-service systems.*
AMS 60K25, 68M20; 12 pp.; **key words:** queueing system, cyclic service, switch-over times, mean waiting time, conservation law.

Abstract: This paper considers single-server, multi-queue systems with cyclic service. Non-zero switch-over times of the server between consecutive queues are assumed. A stochastic decomposition for the amount of work in such systems is obtained. This decomposition allows a short derivation of a 'pseudo-conservation law' for a weighted sum of the mean waiting times at the various queues. Thus several recently proved conservation laws are generalised and explained.

OS-N8602. J.L. van den Berg. *Queueing analysis of a virtual circuit in a computer communication network with window flow control.*
AMS 60K25, 68M20; 53 pp.; **key words:** flow control, virtual circuit, overflow, closed queueing model, throughput, end-to-end delay.

Abstract: This note studies a virtual circuit with finite buffer space in a computer communication network with window flow control. An approximation method is derived for the throughput in this circuit. The method can also be used to approximate other important performance measures. For a special case an exact analysis is presented.

NM-R8605. P.J. van der Houwen & F.W. Wubs. *The method of lines and exponential fitting.*
AMS 65M20, 78B15; 9 pp.; **key words:** numerical analysis, hyperbolic equations, periodic solutions.

Abstract: When the method of lines is used for solving time-dependent partial differential equations, finite differences are commonly employed to obtain the semidiscrete equations. Usually, if the solution is expected to be smooth, symmetric difference formulas are chosen for approximating the spatial derivatives. These difference formulas are almost invariably based on Lagrange type differentiation formulas. However, if it is known in advance that periodic components of given frequency dominate in the solution, more accurate difference formulas, based on exponentials with imaginary exponents, are available. This paper derives such formulas and presents numerical results which clearly indicate that the accuracy can be improved considerably by exploiting additional knowledge on the frequencies of the solution.

NM-R8606. W.H. Hundsdorfer. *A note on monotonicity of a Rosenbrock method.*
AMS 65L20, 65H10; 6 pp.; **key words:** stiff differential equations, monotonicity, Rosenbrock methods, nonlinear algebraic equations, modified methods.

Abstract: For a dissipative differential equation with stationary solution u^*, the difference between any solution $U(t)$ and u^* is nonincreasing with t. In this note we present necessary and sufficient conditions in order for a similar monotonicity property to hold for numerical approximations computed from a Rosenbrock method. Our results also provide global convergence results for some modifications of Newton's method.

NM-R8607. K. Burrage & W.H. Hundsdorfer. *The order of B-convergence of algebraically stable Runge-Kutta methods.*
AMS 65L05, 7 pp.; **key words:** numerical analysis, stiff initial value problems, implicit Runge-Kutta methods, *B*-convergence.

Abstract: In a paper in Computing we have shown that for a class of semi-linear problems many high order Runge-Kutta methods have order of optimal *B*-convergence one higher than the stage order. In this paper we show that for the more general class of nonlinear dissipative problems such a result holds only for a small class of Runge-Kutta methods and that such methods have at most classical order 3.

NM-R8608. J.M. Sanz-Serna & J.G. Verwer. *Convergence analysis of one-step schemes in the method of lines.*
AMS 65X02, 65M10, 65M20; CR 5.17; 12 pp.; **key words:** numerical analysis, initial boundary value problems in partial differential equations, method of lines, Runge-Kutta schemes, convergence analysis, order reduction.

Abstract: We present an expository account of some fundamental results concerning the analysis of one-step schemes for semidiscretizations of evolutionary problems in partial differential equations. In the paper the emphasis lies on the interplay between the stability and convergence properties of the fully discrete scheme and those of the ordinary differential equations solver. Much attention is paid to the phenomenon of order reduction.

NM-R8609. H.J.J. te Riele. *On the sign of the difference* $\pi(x) - li(x)$.
AMS 11A41, 11M26, 11Y99, 65G99; 6 pp.; **key words:** prime counting function, approximation, sign changes, Riemann hypothesis, zeros of the Riemann zeta function.

Abstract: Let $\pi(x)$ be the number of primes $\leqslant x$ and $li(x) = \int_0^x (\log t)^{-1}\, dt$. It is well known that $\pi(x) \sim li(x)$ as $x \to \infty$ and also that $\pi(x) - li(x)$ changes sign infinitely often. However only negative values of $\pi(x) - li(x)$ have ever been actually computed. Following a method of Sherman Lehman we show that between 6.62×10^{370} and 6.69×10^{370} there are more than 10^{180} successive integers x for which $\pi(x) - li(x) > 0$. This brings down Sherman Lehman's bound on the smallest number x for which $\pi(x) - li(x) > 0$, namely from 1.65×10^{1165} to 6.69×10^{370}. Our result is based on the knowledge of the truth of the Riemann hypothesis for the complex zeros $\beta + i\gamma$ of the Riemann zeta function which satisfy $|\gamma| < 450{,}000$, and on the knowledge of the first 15,000 complex zeros to about 28 digits and the next 35,000 to about 14 digits.

NM-R8610. B.P. Sommeijer. *NUMVEC FORTRAN Library manual. Chapter: Parabolic PDEs; Routine:BDMG.*
AMS 65M20, 65L05, 65V05; CR 5.17; 16 pp.; **key words:** software, parabolic differential equations, multigrid methods, nonlinear Chebeyshev iteration.

Abstract: This document describes the NUMVEC FORTRAN Library routine BDMG, which integrates in time a semidiscrete scalar parabolic partial differential equation defined over a two-dimensional rectangular region. The time integration is based on the second-order Backward

Differentiation method; the resulting implicit relations are solved by employing a Multi Grid technique.

NM-R8611. S.P. Spekreijse. *Multigrid solution of monotone second-order discretizations of hyperbolic conservation laws.*
AMS 35L65, 65N05, 76G15; 19 pp.; **key words:** conservation laws, multigrid methods.
Abstract: This paper is concerned with two subjects: the construction of second-order accurate monotone upwind schemes for hyperbolic conservation laws and the multigrid solution of the resulting discrete steady state equations. By the use of an appropriate definition of monotonicity, it is shown that there is no conflict between second-order accuracy and monotonicity (neither in one nor in more dimensions). It is shown that a symmetric block Gauss-Seidel underrelaxation (each block is associated with 4 cells) has satisfactory smoothing rates. The success of this relaxation is due to the fact that, by coupling the unknowns in such blocks, the nine-point stencil of a second-order 2D upwind discretization changes into a five-point block stencil.

NM-R8612. P.J. van der Houwen & B.P. Sommeijer. *Phase-lag analysis of implicit Runge-Kutta methods.*
AMS 65L05, CR G.1.7, G.1.8; 16 pp.; **key words:** numerical analysis, ordinary differential equations, Runge-Kutta methods, periodic solutions.
Abstract: We analyse the phase errors introduced by implicit Runge-Kutta methods when a linear inhomogeneous test equation is integrated. It is shown that the homogeneous phase errors dominate if long interval integrations are performed. Homogeneous dispersion relations for the special class of DIRK methods are derived and a few high-order dispersive DIRK methods are constructed. These methods are applied to systems of linear differential equations with oscillating solutions and compared with the 'conventional' DIRK methods of Nørsett and Crouzeix.

NM-R8613. E. de Goede. *Stabilization of the Lax-Wendroff methods and a generalized one-step Runge-Kutta method for hyperbolic initial-value problems.*
AMS 65M10, 65M20; 12 pp.; **key words:** stabilization, hyperbolic equations, smoothing operators.
Abstract: In order to integrate hyperbolic systems explicit time integrators are specialley appropriate. Implicit methods allow large integration steps, but require more storage and are more difficult to implement than explicit methods. However, explicit methods are subject to a restriction on the integration step. This restriction is a drawback if the variation of the solution in time is so small that accuracy considerations would allow a larger integration step. In this report we apply a smoothing technique in order to stabilize the Lax-Wendroff method and a generalized one-step Runge-Kutta method. Using this technique, the integration step is not limited by stability considerations.

NM-R8614. W. Hoffmann & W.M. Lioen. *NUMVEC FORTRAN Library manual. Chapter: simultaneous linear equations.*
AMS 65V05, 65F05, 15A06; CR 5.14; 33 pp.; **key words:** Gaussian elimination, LDU-decomposition, linear equations, software.
Abstract: This document describes a set of NUMVEC FORTRAN Library routines, dealing with the unique solution of real linear systems. Presently, only highly optimized non-portable implementations for the CYBER 200 series computer systems are included in the Library.

NM-R8615. J.H.M. ten Thije Boonkkamp. *The odd-even hopscotch pressure*

correction scheme for the incompressible Navier-Stokes equations.
AMS 65M20, 76D05; 17 pp.; **key words:** Navier-Stokes equations, odd-even hopscotch method, pressure correction method.

Abstract: The odd-even hopscotch (OEH) scheme is a time-integration technique for time-dependent partial differential equations. In this paper we apply the OEH scheme to the incompressible Navier-Stokes equations in conservative form. In order to decouple the computation of the velocity and the pressure, the OEH scheme is applied in combination with the pressure correction technique. The resulting scheme is referred to as the odd-even hopscotch pressure correction (OEH-PC) scheme. This scheme requires per time step the solution of a Poisson equation for the computation of the pressure. For space discretization we use standard central differences. We applied the OEH-PC scheme to the Navier-Stokes equations for the computation of an exact solution, with the purpose of testing the (order of) accuracy of the scheme in time as well as in space. It turned out that the OEH-PC scheme behaves as a second order scheme in time and space. Furthermore we applied the OEH-PC scheme for the computation of the flow in a glass furnace. Finally, a comparison between two Poisson solvers for the computation of the pressure is presented.

NM-R8616. B. Koren. *Evaluation of second order schemes and defect correction for the multigrid computation of airfoil flows with the steady Euler equations.*
AMS 65N30, 76G15, 76H05; 17 pp.; **key words:** steady Euler equations, second order schemes, defect correction, multigrid methods.

Abstract: Second order accurate Euler flow solutions are presented for some standard airfoil test cases. Second order accuracy is obtained by a defect correction process. Several schemes are considered for the computation of the second order defect. In each defect correction cycle, the solution is computed by a non-linear multigrid iteration, in which Collective Symmetric Gauss-Seidel relaxation is used as smoothing procedure. A finite volume Osher discretization is applied. The computational method does not require tuning of parameters. The solutions obtained show a good resolution of all flow phenomena, and are obtained at low computational costs. The rate of convergence is grid-independent. The method contributes to the state of the art in efficiently computing airfoil flows with discontinuities.

NM-R8617. P.J. van der Houwen, B.P. Sommeijer & F.W. Wubs. *Analysis of smoothing operators in the solution of partial differential equations by explicit difference schemes.*
AMS 65M10, 65M20; CR 5.17; 18 pp.; **key words:** numerical analysis, initial boundary value problems in partial differential equations, method of lines, explicit integration methods, smoothing, stability.

Abstract: A smoothing technique for the 'preconditioning' of the right-hand side of semi-discrete partial differential equations is analysed. For a parabolic and a hyperbolic model problem optimal smoothing matrices are constructed which result in a substantial amplification of the maximal stable integration step of arbitrary explicit time integrators when applied to the smoothed problem. This smoothing procedure is illustrated by integrating both linear and nonlinear parabolic and hyperbolic problems. The results show that the stability behaviour is comparable with that of the Crank-Nicholson method; furthermore, if the problem belongs to the problem class in which the time derivative of the solution is a smooth function of the space variables, then the accuracy is also comparable with that of the Crank-Nicholson method.

NM-R8618. J.G. Blom & H. Brunner. *Discretized collocation and iterated collocation for nonlinear Volterra integral equations of the second kind.*

AMS 45-04, 65R20, 45D05, 45L10; 18 pp.; **key words:** systems of nonlinear Volterra integral equations of the second kind, approximation by polynomial splines, (iterated) collocation, local superconvergence, error estimations, variable stepsize, numerical methods, computer software.

Abstract: In this paper a FORTRAN code is described for the approximate solution of systems of nonlinear Volterra integral equations of the second kind. The algorithm is based on polynomial spline collocation, possibly in combination with the corresponding iterated collocation. It exploits certain local superconvergence properties for the error estimation and the stepsize strategy.

NM-N8601. E. de Goede. *A comparison of finite difference schemes for the numerical solution of hyperbolic equations.* (In Dutch)

AMS 65M10; 10 pp.; **key words:** hyperbolic equations, finite difference schemes, stability, efficiency.

Abstract: In this note we compare existing finite difference schemes for first order hyperbolic equations in two space dimensions. The stability and efficiency of the difference schemes is examined.

MS-R8602. S.A. van de Geer. *A new approach to least squares estimation, with applications.*

AMS 60B10, 60G50, 62J05; 5 pp.; **key words:** consistency, entropy, empirical measure, uniform convergence.

Abstract: The regression model $\mathbf{y}=g(\mathbf{x})+\epsilon$ and least squares estimation are studied in a general context. By making use of empirical process theory, it is shown that the essential condition for L^2-consistency of the least squares estimator $\hat{\mathbf{g}}_n$ of g is an entropy condition on the class $\mathcal{G}$ of possible regression functions. This result is applied in parametric and nonparametric regression.

MS-R8603. R.D. Gill & J.A. Wellner. *Large sample theory of empirical distributions in biased sampling models.*

AMS 62G05, 60F05, 62G30, 60G44; 22 pp.; **key words:** selection bias models, Vardi's estimator, nonparametric Maximum Likelihood.

Abstract: Vardi (1985) introduced an s-sample model for biased sampling, gave conditions which guarantee the existence and uniqueness of the nonparametric maximum likelihood estimator $\mathbb{G}_n$ of the common underlying distribution G, and discussed numerical methods for calculating the estimator. Here we examine the large sample behaviour of the NPMLE $\mathbb{G}_n$, including results on uniform consistency of $\mathbb{G}_n$, convergence of $\sqrt{n}(\mathbb{G}_n - G)$ to a Gaussian process, and asymptotic efficiency of $\mathbb{G}_n$, as an estimator of G. The proofs are based upon recent results for empirical processes indexed by sets and functions, properties of irreducible M-matrices, and the homotopy invariance theorem. A final section discusses examples and applications to stratified sampling, 'choice-based' sampling in econometrics, and 'case-control' studies in biostatistics.

MS-R8604. R.D. Gill. *Non- and semi-parametric maximum likelihood estimators and the von Mises method* (Part 1).

AMS 62G05, 62G20, 60B12, 60F17, 46A05; 22 pp.; **key words:** non-parametric maximum likelihood, von Mises method, compact differentiation, Hadamard differentiation, asymptotically efficient estimation.

Abstract: After introducing the approach to von Mises derivatives based on compact differentiation due to Reeds (1976), we show how non-parametric maximum likelihood estimators can often be defined by solving infinite dimensional score equations. Each component of the score equation corresponds to the derivative of the log likelihood for a one-dimensional parametric

submodel. By means of examples we show that it usually is not possible to base consistency and asymptotic normality theorems on the implicit function theorem. However (in Part II) we show for a particular class of models, that once consistency (in a rather strong sense) has been established by other means, asymptotic normality and efficiency of the non-parametric maximum likelihood estimator can be established by the von Mises method.

AM-R8602. H. Roozen. *Equilibrium and extinction in stochastic population dynamics.*
AMS 35A40, 35B40, 35R60, 92A15; 22 pp.; **key words:** ray method, confidence region, extinction time, generalized Lotka-Volterra system, numerical simulation of stochastic birth-death processes.
Abstract: Stochastic models of interacting biological populations, with birth and death rates depending on the population size, are studied in the quasi-stationary state. Confidence regions in the state space are constructed by a new method for the numerical solution of the ray equations. The concept of extinction time, which is closely related to the concept of stability for stochastic systems, is discussed. Results of numerical calculations for two-dimensional stochastic population models are presented.

AM-R8603. H.J.A.M. Heijmans & J.A.J. Metz. *Small parameters in structured population models and the Trotter-Kato theorem.*
AMS 92A15, 35A35, 47D05; 19 pp.; **key words:** structured population, limit transition, C_0-semigroup, Trotter-Kato theorem.
Abstract: In this paper we discuss by means of two examples the justification of some (often implicit) limit arguments used in the development of structured population models. The first example considers the usual equation for size dependent population growth, in which it is implicitly assumed that discrete finitely sized young are produced from infinitesimal contributions by all potential parents. The second example shows how a pair of sink-source terms may transform into a side condition relating the appearance of individuals in the interior of the individual state space to the outflow of individuals at its boundary. The main mathematical tool for dealing with these example is the Trotter-Kato theorem.

AM-R8604. O. Diekmann. *Perturbed dual semigroups and delay equations.*
AMS 47D05, 34K05; 8 pp.; **key words:** C_0-semigroups, weak * continuous semigroups, dual semigroups, bounded perturbation of the generator, variation-of-constants formula, retarded functional differential equations, delay equations.
Abstract: The theory of dual semigroups on non-reflexive Banach spaces can be used to define a natural generalization of the notion of a bounded perturbation of the generator and a new version of the variation-of-constants formula. This approach was developed in joint work with Ph. Clément, M. Gyllenberg, H.J.A.M. Heijmans and H.R. Thieme, motivated by some applications to physiologically structured population growth models. In this paper it is shown that delay differential equations fit very well into exactly the same functional analytic framework.

PM-R8602. G.F. Helminck. *Deformations of connections, the Riemann-Hilbert problem and τ functions.*
AMS 35Q15, 35F20; 16 pp.; **key words:** Riemann-Hilbert problem, integrable connections, isomonodromic deformation, Fredholm determinant, Grasmann manifold.
Abstract: We give sufficient conditions for the existence of integrable deformations of a rational

linear ODE on the projective line and we show when the related connection form obtains a reduced form. Certain coefficients in this reduced form admit a meromorphic continuation to the whole parameter space, while their poles coincide with the zero-set of a Fredholm determinant τ. These properties are similar to the ones holding for the solutions of the KP-hierarchy and form a generalization of work by Malgrange.

CWI Activities

Summer 1986

With each activity we mention its frequency and (between parentheses) a contact person at CWI. Sometimes some additional information is supplied, such as the location if the activity will not take place at CWI.

Study group on Analysis on Lie groups. Jointly with University of Leiden. Biweekly. (T.H. Koornwinder)

Seminar on Integrable Systems Theory. Once a month. (M. Hazewinkel)
A central object of study will be the work of Belarin and Drinfeld, especially the relation between simple Lie algebras and solutions of the so-called classical Yang-Baxter equation. Furthermore the quantum-Yang-Baxter equation, the Fundamental Poisson relations, the Fundamental Commuting relations which form the basis of the 'Quantum Inverse Spectral Method' will be discussed.

Seminar on Algebra and Geometry. Once a month. (A.M. Cohen)
Gordan's work on covariants of $SL_2(\mathbb{C})$ (J. Brinkhuis). Mumford's construction of an algebraic surface resembling $\mathbb{C}^2$ (M. van der Put). The geometry of subgroups of order 3 in certain finite groups (F.G.M.T. Cuypers).

Cryptography working group. Monthly. (J.H. Evertse)

Colloquium 'STZ' on System Theory, Applied and Pure Mathematics. Twice a month. (J. de Vries)

Study group 'Biomathematics'. Lectures by visitors or members of the group. Jointly with University of Leiden. Bimonthly (O. Diekmann)
Topics for the next meetings are: *stochastic population dynamics, dynamics of structured populations.*

Study group on Nonlinear Analysis. Lectures by visitors or members of the group. Jointly with University of Leiden. Bimonthly (O. Diekmann) The purpose is: to follow and investigate recent developments on qualitative analysis of nonlinear equations; to stimulate and support the research of the participants.

Progress meetings of the Applied Mathematics Department. Weekly (N.M. Temme)
New results and open problems on the research topics of the department: biomathematics, mathematical physics, asymptotic and applied analysis, image analysis.

Colloquium Image Processing: Theory and Practice. Applied Mathematics and Mathematical Statistics. 9, 30 October, 20 November, 11 December. (H.J.A.M. Heijmans)
The following topics will be treated:
Applications of image processing and analysis in medical science, biology, physics, astronomy, industry, etc. Mathematical aspects as transformation techniques, filtering techniques, statistical analysis, topology, etc.; Role of the computer: hardware and software, architecture of image computers, etc. Related topics: laser optics (holography), visual observations by human beings and animals, etc. Long term planning, prospects, etc.

Study group on Statistical and Mathematical Image Analysis. (Parallel with colloquium 'Image Analysis'). Every three weeks. (R.D. Gill)
The group is presently studying J. Serra's approach to image analysis, 'mathematical morphology', and recent statistical contributions using Markov field modelling due to S. and G. Geman, J. Besag and B. Ripley.

Progress meetings of the Mathematical Statistics Department. Biweekly (R. Helmers)
Talks by members of the department on recent developments in research and consultation. Also talks by E. Valkeila (Helsinki) on counting processes.

Lunteren meeting on Stochastics. 17, 18 and 19 November 1986 at 'De Blije Werelt', Lunteren. Jointly with Dutch Statistical and Mathematical Societies. (R. Helmers)
Invited speakers: J. Besag (Durham, U.K.), G.L. O'Brien (Toronto, Canada), D.R. Cox (London, U.K.), C.M. Goldie (Brighton, U.K.), L. Russo (Rome, Italy), E.V. Slud (Maryland, USA).

Seminar 'von Mises and NPMLE'. Jointly with University of Leiden. Weekly. (R.D. Gill)
A rigorous approach to the von Mises calculus (due to J.A. Reeds) based on Hadamard differentiability is used to derive properties of nonparametric maximum likelihood estimators.

Study group on Combinatorial Optimization. Biweekly. (B.J. Lageweg)

System Theory Days. Irregular. (J.H. van Schuppen, J.M. Schumacher)

Study group on System Theory. Biweekly. (J.M. Schumacher)

Colloquium on Queueing Theory and Performance Evaluation. Irregular. (O.J. Boxma)

Conference on Numerical Mathematics. 29 September - 1 October 1986 at

Zeist. (J.G. Verwer)

Invited speakers: M.J. Baines (University of Reading, England), P.R. Eiseman (Columbia University, New York, USA), N.P. Weatherill (Aircraft Research Association Ltd, Bedford, England), M. Crouzeix (University of Rennes, France), J.M. Sanz-Serna (University of Valladolid, Spain), J.C. Butcher (University of Auckland, New Zealand).

Progress meetings on Numerical Mathematics. Weekly. (H.J.J. te Riele)

Study group on Numerical Software for Vector Computers. Monthly. (H.J.J. te Riele)

Study group on Differential and Integral Equations. Lectures by visitors or group members. Irregular. (H.J.J. te Riele)

Study group on Graphics Standards. Monthly. (M. Bakker)

National Study Group on Concurrency. Jointly with Universities of Leiden and Eindhoven. 26 September, 24 October, 21 November and 12 December. (J.W. de Bakker)

Study group on Dialogue Programming. (P.J.W. ten Hagen)

Study group on Logical Aspects on Artificial Intelligence. Biweekly. (M.L. Kersten & P.J.F. Lucas)

In this study group recent developments in formal theories in artificial intelligence are discussed. The main topics are: knowledge representation, inference methods, non-standard logics and plausible reasoning.

Post-academic Course on PROLOG. 20-21 October. (P.J.F. Lucas)

In this course, both the theoretical foundations of logic programming and the applications of the programming language PROLOG are discussed. The course is meant for researchers and engineers who consider using PROLOG in their projects.

Process Algebra Meeting. Weekly. (J.W. Klop)

Post-academic Course on Modern Techniques in Software Engineering. 25-26 September, 9-10 October, 27-28 November, 11-12 December. (J.C. van Vliet)

Various lecturers present modern techniques and methods for the construction of complex software systems. The course is meant for persons with a background in computer science, who are or will be actively involved in the construction of those systems.

Meeting of the ESPRIT project 348 (Generating of Interactive Programming Environments - GIPE). 20-21 October. (P. Klint)

Visitors to CWI from Abroad

J.Ph. Anker (University of Lausanne, Switzerland) 11-12 April. K.R. Apt (L.I.T.P., Université Paris VII, France) 21 April - 30 May. D.D. Boos (North Carolina State University, USA) 4 June. L. de Branges (Purdue University of Lafayette, USA) 16-18 June. P.E. Caines (McGill University, Montreal, Canada) 21 April. J.A. van Casteren (Universitaire Instelling Antwerpen, Belgium) 28-29 April. L. Corwin (Rutgers University, New Brunswick, USA) 11-12 April. J. Dongarra (Argonne National Laboratory, USA) 30 May. W.H. Enright (University of Toronto, Canada) 29 May. M. Flensted-Jensen (The Royal Veterinary and Agricultural University, Copenhagen, Denmark) 11-12 April. M. Fliess (CNRS, Gif sur Yvette, France) 28-29 April. A. George (University of Waterloo, Canada) 30 May. I. Gertsbakh (Ben Gurion University of the Negev, Beersheva, Israel) 23-24 June. P. Green (University of Durham, England) 26-30 May. B. Harsoyo (University of Jakarta, Indonesia) 23 June - 23 Augustus. H. Hirata (Chiba University, Japan) 20 June. N.L. Hjort (Norwegian Computing Centre, Oslo, Norway) 26-30 May. T. Louis (Harvard School of Public Health, Boston, USA) January-June 1986. D. Moore (Purdue University, Lafayette, USA) 21 May. M. Nakashima (RWTH, Aachen, West Germany) 2 June. M. Nevat (LITP, University of Paris VII, France) 9 June. J.-P. Nicolas (University of Limoges, France) 6-7 May. G. Olafsson (University of Göttingen, West Germany) 11-12 April. D. O'Leary (University of Maryland, USA) 30 May. S.J. Prokhovnik (University of New South Wales, Kensington, Australia) 2 May. W. Respondek (Polish Academy of Science, Warsaw, Poland) 14 May. W.A. Rosenkrantz (University of Massachusetts, USA) 14-16 April. D. Siegmund (University of Stanford, USA) 14 May. J. Smith (John Hopkins University, USA) September 1985 - June 1986. D. Stodolsky (University of Stockholm, Sweden) 1-11 April. F. Sullivan (National Bureau of Standards, Gaithersburg, USA) 30 May. H.

Suzuki (University of Tokyo, Japan) 16 May. H. Takagi (IBM Japan Science Institute, Tokyo, Japan) 7-13 June. S. Ushiki (Kyoto University, Japan) 9 June. J.I. Zucker (University of Buffalo, USA) 21 July - 9 August.

Order Form for CWI Publications

Sales Department
Centre for Mathematics and Computer Science
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

☐ Please send the publications marked below on an exchange basis
☐ Please send the publications marked below with an invoice

	Publication code	Price per copy	Number of copies wanted
☐	CWI Tract 22 *)	Dfl. 45.40	....
☐	CWI Tract 23 *)	26.70	....
☐	CWI Tract 24 *)	12.70	....
☐	CWI Tract 25 *)	12.70	....
☐	CWI Tract 27 *)	20.30	....
☐	CWI Syllabus 10 *)	25.30	....
☐	CS-R8610	4.--	....
☐	CS-R8611	4.--	....
☐	CS-R8612	4.--	....
☐	CS-R8613	4.--	....
☐	CS-R8614	4.--	....
☐	CS-R8615	4.--	....
☐	CS-R8616	4.--	....
☐	CS-R8617	6.40	....
☐	CS-R8619	8.40	....
☐	CS-R8620	49.20	....
☐	CS-R8621	6.40	....

*) not available on exchange

	Publication code	Price per copy	Number of copies wanted
☐	CS-R8622	4.--	....
☐	CS-N8601	4.--	....
☐	CS-N8602	4.--	....
☐	CS-N8603	4.--	....
☐	CS-N8604	7.70	....
☐	CS-N8605	5.20	....
☐	CS-N8606	10.20	....
☐	OS-R8604	4.--	....
☐	OS-R8605	4.--	....
☐	OS-R8606	4.--	....
☐	OS-N8602	8.90	....
☐	NM-R8605	4.--	....
☐	NM-R8606	4.--	....
☐	NM-R8607	4.--	....
☐	NM-R8608	4.--	....
☐	NM-R8609	4.--	....
☐	NM-R8610	4.--	....
☐	NM-R8611	4.--	....
☐	NM-R8612	4.--	....
☐	NM-R8613	4.--	....
☐	NM-R8614	5.20	....
☐	NM-R8615	4.--	....
☐	NM-R8616	4.--	....
☐	NM-R8617	4.--	....
☐	NM-R8618	4.--	....
☐	NM-N8601	4.--	....
☐	MS-R8602	4.--	....
☐	MS-R8603	4.--	....
☐	MS-R8604	4.--	....
☐	AM-R8602	4.--	....
☐	AM-R8603	4.--	....
☐	AM-R8604	4.--	....
☐	PM-R8602	4.--	....

If you wish to order any of the above publications please tick the appropriate boxes and return the completed form to our Sales Department.
Don't forget to add your name and address!
Prices are given in Dutch guilders and are subject to change without notice. Foreign payments are subject to a surcharge per remittance to cover bank, postal and handling charges.

Name ....

Street ....

City ....

Country ....

Signature ....

Date ....

Contents