Available online at www.sciencedirect.com
SCIENCE DIRECT®
Parallel Computing 31 (2005) 877–899
PARALLEL COMPUTING
www.elsevier.com/locate/parco

Predicting the execution times of parallel-independent programs using Pearson distributions

G.L. Reijns, A.J.C. van Gemund *

Department of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, P.O. Box 5031, NL-2600 GA Delft, The Netherlands

Received 15 November 2004; received in revised form 4 April 2005; accepted 6 June 2005
Available online 9 August 2005

Abstract

Predicting the execution time of parallel programs involves computing the maximum or minimum of the execution times of the tasks involved in the parallel computation. We present a method to accurately compute the distribution of the largest (Max) and the smallest (Min) execution time of the composite of a number of parallel programming tasks, each having an independent, stochastic, arbitrary workload. The Max function applies to the general case that the composite task completes at the time its longest constituent task terminates. The Min function applies when the completion of its shortest task terminates the whole parallel process, such as in a parallel searching program. Both the Min and Max density function of a constituent task are characterized in terms of a Pearson distribution. Due to its accuracy, the presented method is especially of interest when the performance of time critical parallel applications must be derived. Both prediction methods are tested against three well-known distributions. Furthermore, the Max prediction method is also tested against a number of measured real-life data parallel programs with different degree of parallelism. The results show excellent accuracy of better than 1% with a very few exceptions in extreme situations.

* Corresponding author. Tel.: +31 15 278 6411; fax: +31 15 278 4898.
E-mail address: a.j.c.vangemund@ewi.tudelft.nl (A.J.C. van Gemund).

0167-8191/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.parco.2005.06.003

Keywords: Performance prediction; Stochastic graphs; Pearson distribution; Order statistics

1. Introduction

The research in this paper is primarily aimed at predicting the execution time of a set of time critical parallel programs. For that reason a major part of this paper is devoted to the subject of performance analysis of a set of parallel tasks in which the composite program completes at the time its longest constituent task terminates (Max procedure). However there are also parallel programs in which the completion of its shortest task determines the termination of the composite set e.g., in seach programs (Min procedure). We present the results and accuracies obtained with both the Max and Min procedures and provide also data on real-life parallel programs using the Max procedure.

Performance prediction is a valuable engineering tool in the design of parallel programs. Usually the metric of concern is the (mean) execution time. Due to a program's input data set dependency, however, execution times tend to be stochastic, reflecting the variety of the input data spectrum across multiple runs (for examples, see Sections 6.3.1 and 6.3.2). Consequently, the preferred solution of a performance model is distribution information, rather than a single scalar. Especially for time-critical applications, distribution information is paramount, e.g., in order to predict the probability a program will exceed a certain execution time threshold. In this paper we consider the Max and Min performance prediction problem where a parallel program is modeled in terms of a stochastic task graph, where vertices represent tasks and edges represent task precedence relations.

The performance prediction of stochastic task graphs has been subject of extensive research [9,12,14–16,20,21]. Research has often focussed on series–parallel graphs because of their favorable properties. The series construction of a stochastic task graph is rather easy to analyze, as the moments of the composite task can directly be expressed in the moments of the constituent tasks [9,10]. In contrast, the parallel construction is much more difficult to handle. The Max and Min construction requires the use of the density distributions of the execution times of the constituent tasks. An approximation of the density function can be obtained from the (given or measured) moments of the execution time. The most frequently applied technique in this field assumes negative exponential distributions of the task execution times, which entails a Markov or Mean-Value Analysis. While the approximation by a negative exponential distribution is adequate for a number of applications, it does not suffice when handling time critical parallel programs. In time-critical programs it is important to predict the upper bound of the execution time distribution (for, e.g., scheduling purposes) rather than just a mean value, such as, e.g., the 99th percentile, so one can predict the execution time that only 1% of the program runs will exceed. A prediction based on the negative exponential distribution will produce a very large error (e.g., more than 100% [9,10] when predicting a parallel section of 128 tasks, each having a normal distribution, a distribution that is much closer to what one measures in real programs than negative exponential distributions [9,10]).

Due to its accuracy we have chosen to characterize the density function of an arbitrary distribution of the execution time of a basic task by the Pearson distribution which uses the first four moments. The 4-parameter Pearson distribution provides a much better, and versatile model than, e.g., the single-parameter negative exponential distribution [9,10]. Measurements and simulations have been carried out in order to test the validity of our approach. The remainder of the paper is organized as follows. In Section 2 an analysis of the parallel composition is given using the Max and Min construction. In Section 3 we review related work. In Section 4 the Pearson method with our adaptations is described and its coverage. In Section 5 our method to derive the Max and Min composite task is given, using numerical integration, while in Section 6 analytical and measurement results are compared. Conclusions are given in Section 7.

2. Analysis of parallel composition

Let us consider a parallel composition of n processing tasks having execution times X_i, $i = 1,\ldots,n$. The resulting execution time X of the parallel Max composition is given by:

$$X = \max_{i=1,\ldots,n} X_i. \tag{1}$$

Many authors have assumed the X_i to be deterministic [1,2,4,5,7,17,18,22,24]. While Eq. (1) indeed yields a correct prediction when X_i are deterministic, interpreting Eq. (1) in terms of mean values when X_i are stochastic introduces errors. An exact solution for the distribution of X can be obtained using the cumulative density functions (cdf) of its constituent components. Let $\overline{F_i}(x)$ represent the cdf of X_i and $\overline{F}(x)$ that of X. For independent X_i it follows that [26]

$$\overline{F}(x) = \overline{F}_1(x) \cdot \overline{F}_2(x) \cdot \cdots \cdot \overline{F}_n(x). \tag{2}$$

In the case the composite program completes at the time its shortest constituent task terminates the Min function is used. An exact solution of the distribution of

$$X = \min_{i=1,\ldots,n} X_i \tag{3}$$

can be obtained by:

$$1 - \overline{F}(x) = \{1 - \overline{F}_1(x)\} \cdot \{1 - \overline{F}_2(x)\} \cdot \cdots \cdot \{1 - \overline{F}_n(x)\}. \tag{4}$$

In order to predict the execution time distribution of a parallel task Max composition we must derive the four moments of the composite function $F(x)$. Differentiating Eq. (2) when all constituent tasks have the same stochastic properties (i.e., $F_i = F_1$) results in:

$$f(x) = \frac{\mathrm{d}F_1^n(x)}{\mathrm{d}x} = nf_1(x)F_1^{n-1}(x) \tag{5}$$

and the moment k:

$$\mathsf{E}(X^k) = n \int_{-\infty}^{+\infty} x^k f_1(x) F_1^{n-1}(x)\,\mathrm{d}x. \tag{6}$$

Differentiating Eq. (4) when all constituent tasks have the same stochastic properties results in:

$$f(x) = n f_1(x)\{1 - F_1(x)\}^{n-1} \tag{7}$$

and the moment k:

$$\mathsf{E}(X^k) = n \int_{-\infty}^{+\infty} x^k f_1(x)\{1 - F_1(x)\}^{n-1}\,\mathrm{d}x. \tag{8}$$

After we have found an equation for $f_1(x)$ for which the Pearson approximation was taken (Section 4), Eqs. (5) and (6), or (7) and (8) are solved using numerical integration. The moments of a composite task consisting of two different tasks can be derived in a very similar, though a little more complex manner.

3. Related work

A number of methods have been reported to derive numerical as well as analytical solutions for the Max parallel processing problem. The Max problem is now first discussed followed by some remarks on the Min problem. For general symmetric distributions, approximate analytical solutions have been reported [12,16]. The following approximation is valid for such distributions in the case of n parallel tasks with the same stochastic properties [12]:

$$\mathsf{E}[X] \approx \mathsf{E}[X_1] + \sigma[X_1]\sqrt{2\log(0.4n)}$$

in which $\sigma[X_1]$ is the standard deviation of X_1. The main problem of this approximation is its restriction to symmetrical distributions: in practice, most of the distributions are found to be asymmetrical. Besides, only the first moment can be derived. Attempts have been reported where the distribution $F_1(x)$ is approximated by analytic functions which enable the integration problem in Eq. (6) to be solved analytically. Within this category we recognize the class of exponomial distributions [21] or the class of discrete distributions using the z-transform [14]. However, such resulting distributions are practically never found in real computer programs. In [11] the authors succeeded in solving the n-ary composition problem by first solving the binary one but the computing cost can become excessive. A solution based on the use of the first two moments of a density function [13] was found unsatisfactory (see Section 4.1). The solution presented in [9] approximates $F_1(x)$ by a lambda distribution [19] having four parameters. Knowing the four moments, one obtains the values of these parameters using a table look-up or by applying a linear programming method [19]. The method presented in [9] has the disadvantage in respect to the solution presented in our paper that it is less accurate, requires higher computing cost and cannot be used properly to solve the binary composition with two different tasks.

In [20] the accuracies obtained with real-life programs were based on comparisons with an insufficient number of simulation runs and the paper did neither contain any information on the Min procedure. The solution of the Min problem has received anyhow less attention than that of the Max problem. The authors in [12,15,21] discuss the solution of the Min problem but restrict the analysis either to specific distributions or assume distributions to be symmetric. Examples of specific distributions are the Erlang and Hyper-exponential distributions discussed in [12]. While proven powerful tools for reliability modeling, such workload distributions are rarely found in practice. The solution presented in [8] is less accurate than the one presented in this paper and has the additional disadvantage to be sometimes unstable. The current paper does not assume specific restrictions on distributions and the solution can be applied to both n-ary and binary compositions. Furthermore, computational cost is low despite the use of numerical integration (see Section 5.1).

4. The Pearson distribution

4.1. Justification

Before explaining the Pearson distribution we provide two examples to illustrate that one needs to approximate an arbitrary density function of the execution time of a basic task by using its first four moments in order to obtain the required accuracy for real-time applications. In the first example we consider two different parallel sections, each comprising 64 parallel Max tasks, each task having a density function with zero mean, second moment of 1, fourth moment of 5, but with a third moment of 1.52 in one section, and -1.52 in the other section. When the third moment is positive this results in a mean of the section's execution time of 3.27 with a variance of 11.1, while the other section measures a mean of 0.89 with a variance of 0.79. The second example illustrates that the fourth moment is neither to be neglected. Consider again two different parallel sections each comprising 64 parallel Max tasks, each task having a density function with zero mean, second moment of one, third moment of 0.173, but with a fourth moment of 1.84 in the first section and 5.38 in the other section. This results in a mean of the first section's composite execution time of 1.38 with a variance of 3.37 while the second section measures a mean of 2.77 with a variance of 8.59. Due to its accuracy (Section 6) we have chosen to characterize the density function of any arbitrary distribution of a basic task by the Pearson approximation, which uses the first four moments. In addition, the four Pearson characterization parameters (Section 4.2) can directly be expressed in the four given or measured moments, different from the method presented in [8,9], where a table look-up or linear programming method is required to obtain these parameters. With its four parameters the Pearson distribution can model a wide range of distribution shapes, including standard distributions, such as the negative exponential, Gaussian, and uniform distribution, but also distributions as found in real-life programs, as examplified by well-known kernels and benchmark programs such as PSRS, WATOR, APSP, NAS-EP, Spalsh2-Barnes [10]. As the Pearson distribution and

its mathematics have been described in detail in [6,25], only some of the more important issues are reported here.

4.2. Main details of the Pearson distribution

It has been shown by Pearson [6,25] that the derivative of any density function $\overline{f}(x)$ can be approximated by:

$$\frac{\mathrm{d}\overline{f}(x)}{\mathrm{d}x} = \frac{(x+a)\overline{f}(x)}{d_0 + d_1 x + d_2 x^2} \tag{9}$$

or

$$(d_0 + d_1 x + d_2 x^2)\,\mathrm{d}\overline{f}(x) = (x+a)\overline{f}(x)\,\mathrm{d}x. \tag{10}$$

Calculations can be simplified by shifting the density function $\overline{f}(x)$ along the x-axis until its first moment becomes zero. We call the shifted function the centralized density function $f(x)$. As the shape and therefore its variance, skew and kurtosis do not change by the shift, we will work with the centralized function to determine the parameters a, d_0, d_1, and d_2. This also provides a better accuracy. At the end of the calculations a return shift will then be executed. The relation between the moments m_1, m_2, m_3 and m_4 of the original density function $\overline{f}_1(x)$ and the moments u_1, u_2, u_3 and u_4 of the centralized function $f_1(x)$ are given by [6]:

$$\begin{aligned} u_1 &= 0, \\ u_2 &= m_2 - m_1^2, \\ u_3 &= m_3 - 3m_1 m_2 + 2m_1^3, \\ u_4 &= m_4 - 4m_1 m_3 + 6m_1^2 m_2 - 3m_1^4. \end{aligned} \tag{11}$$

The four equations relating u_1, u_2, u_3, u_4 with a, d_0, d_1, d_2 can be obtained from the centralized Eq. (10) by multiplying each side of this equation by x^p and subsequently putting $p = 0$, $p = 1$, $p = 2$, and $p = 3$, which gives four equations

$$\begin{aligned} a &= (3u_2^2 u_3 + u_3 u_4)/q, \\ d_0 &= (-4u_2^2 u_4 + 3u_2 u_3^2)/q, \\ d_1 &= (-3u_2^2 u_3 - u_3 u_4)/q, \\ d_2 &= (3u_3^2 + 6u_2^3 - 2u_2 u_4)/q, \end{aligned} \tag{12}$$

where $q = -18u_2^3 + 10u_2 u_4 - 12u_3^2$.

Depending on whether the equation

$$d_0 + d_1 x + d_2 x^2 = 0 \tag{13}$$

has real roots with different signs, complex roots or real roots with the same sign, the integration of Eq. (9) results in different analytical expressions [6,25]. More precisely, the analytical expression to be obtained depends on the criterion:

$$\eta = \frac{d_1^2}{4 d_0 d_2}.$$

By defining:

$$\text{skew} = \sqrt{B_1} = \frac{u_3}{u_2^{3/2}} \tag{14}$$

$$\text{kurtosis} = B_2 = \frac{u_4}{u_2^2} \tag{15}$$

η becomes:

$$\eta = \frac{B_1(B_2+3)^2}{4(2B_2-3B_1-6)(4B_2-3B_1)}. \tag{16}$$

It follows that:

- if $\eta < 0$, the solution of Eq. (13) has real roots with different signs.
- if $0 < \eta < 1$, the solution of Eq. (13) has complex roots.
- if $\eta > 1$, the solution of Eq. (13) has real roots with equal sign.

If $\eta = 0$, B_1 must be zero as well as the skew (B_2 cannot become negative). Depending on the value of B_2, this case is treated as Pearson type I or type II. The skew is a measure of the asymmetry of the density function, as indicated on the left-hand side of Fig. 1. When the skew is zero, the density function is symmetric with respect to its mean. The kurtosis represents the peakedness of the density function (right figure).

4.3. Pearson distribution of the first main type

If Eq. (13) has real roots with different signs, centralized Eq. (9) can be written as:

$$\frac{df(x)}{dx} = \frac{(x+a)f(x)}{d_0+d_1x+d_2x^2} = \frac{(a_1-a)f(x)}{d_2(a_1+a_2)(x+a_1)} + \frac{(a_2+a)f(x)}{d_2(a_1+a_2)(a_2-x)}.$$

After integration this results in

$$f(x) = K_1(x+a_1)^A(a_2-x)^B, \tag{17}$$

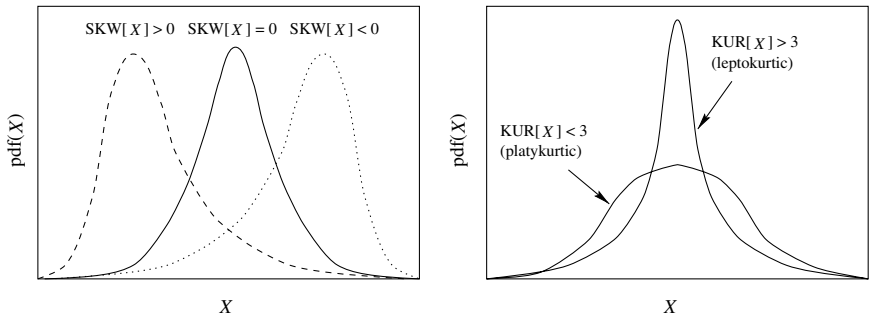

Fig. 1. Shape of density functions.

where

$$a_1 = \frac{d_1 + \sqrt{d_1^2 - 4d_0d_2}}{2d_2},$$
$$a_2 = \frac{-d_1 + \sqrt{d_1^2 - 4d_0d_2}}{2d_2},$$
$$A = \frac{a_1 - a}{d_2(a_1 + a_2)},$$
$$B = \frac{a_2 + a}{d_2(a_1 + a_2)}$$

and K_1 is a constant. Eq. (17) is only valid for $-a_1 < x < a_2$, outside this domain $f(x) = 0$. It turned out that for negative η values which are very close to zero, the factors A and B can become rather large, which means that $(a_1 + x)^A$ and $(a_2 + x)^B$ can become very large or very small. To avoid that $f(x)$ exceeds the positive limit defined by the use of the IEEE double-precision floating-point notation, we have changed both terms of Eq. (17) to:

$$f(x) = c_1 \left[\frac{2(a_1 + x)}{a_1 + a_2}\right]^A \left[\frac{2(a_2 - x)}{a_1 + a_2}\right]^B. \tag{18}$$

We have used a numerical integration to determine c_1 according to

$$\sum f(x_j) = \frac{1}{c_1}$$

with a summation over all samples between $-a_1$ and a_2. Eq. (18) is valid both for positive and negative values of the skew.

4.4. Pearson distribution of the second main type

The roots of Eq. (13) are now complex and $f(x)$ can be expressed as:

$$f(x) = c_2 \left[1 + \left(\frac{x}{c} - \frac{v}{r}\right)^2\right]^{-m} e^{-v \arctan\left(\frac{x}{c} - \frac{v}{r}\right)}, \tag{19}$$

where

$$m = -\frac{1}{2d_2},$$
$$r = 2m - 2,$$
$$c = \frac{1}{4}\sqrt{u_2(16(r-1) - B_1(r-2)^2},$$
$$v = \pm r(r-2)\frac{\sqrt{B_1u_2}}{4c}$$

with v having the opposite sign of the skew. B_1 has been defined in Section 4.1. Note that the function $f(x)$ stretches out from $-\infty$ to ∞ in principle. In practice we have

restricted these bounds from $-15\sqrt{u_2}$ to $15\sqrt{u_2}$, in which $\sqrt{u_2}$ is the standard deviation. The value of $\pm 15\sqrt{u_2}$ was selected in order to adequately cover the left and the right hand tails of this type of distribution. The constant c_2 is determined by the summation of all samples of Eq. (19) between $-15\sqrt{u_2}$ and $15\sqrt{u_2}$ according to

$$\sum f(x_j) = \frac{1}{c_2}.$$

4.5. Pearson distribution of the third main type

The roots of Eq. (13) are now real but of equal sign. $f(x)$ is expressed as:

$$f(x) = c_3(a_1 + x)^A(a_2 + x)^B, \tag{20}$$

where

$$a_1 = \frac{d_1 - \sqrt{d_1^2 - 4d_0d_2}}{2d_2},$$
$$a_2 = \frac{d_1 + \sqrt{d_1^2 - 4d_0d_2}}{2d_2},$$
$$A = \frac{-a_1 + a}{d_2(-a_1 + a_2)},$$
$$B = \frac{a_2 - a}{d_2(-a_1 + a_2)}.$$

The following properties are valid: A is negative, $|A| > B$, and $a_1 > a_2$. Eq. (20) is valid for x between $-a_2 < x < \infty$. In practice we have replaced ∞ by $15\sqrt{u_2}$. The value of $15\sqrt{u_2}$ was selected in order to adequately cover the right hand tail of this type of distribution.

Again, to avoid that $f(x)$ exceeds a positive limit, Eq. (20) has been modified to Eq. (21). The constant c_3 is determined using a summation over all samples between $-a_2$ and $15\sqrt{u_2}$.

$$f(x) = c_3\left(\frac{a_1 + x}{a_1 - a_2}\right)^A\left(\frac{2(a_2 + x)}{a_2 + 15\sqrt{u_2}}\right)^B. \tag{21}$$

Eq. (21) is valid if the skew is positive. For negative values of the skew, Eq. (21) changes into

$$f(x) = c_4\left(\frac{2(a_1 + x)}{-15\sqrt{u_2} + a_1}\right)^A \cdot \left(\frac{a_2 + x}{a_2 - a_1}\right)^B, \quad -15\sqrt{u_2} < x < -a_1. \tag{22}$$

4.6. Coverage by the Pearson method

If $\eta = 0$ in Eq. (16), then $B_1 = 0$, which means that the skewness is zero and that $f_1(x)$ is symmetrical around the x-axis. The shape of a centralized density function is

Table 1
Valid $B_1 - B_2$ domain

B_1	I lower	I upper	III lower	III upper	II lower	II upper 2% err	II upper 10% err
0.0000	1.80[a]	2.990	–	–	3.000[a]	5.38	7.20
0.0001	1.81	2.994	–	–	3.001	5.38	7.20
0.0003	1.81	2.996	–	–	3.002	5.38	7.20
0.0010	1.81	2.998	–	–	3.004	5.38	7.20
0.0030	1.81	3.000	–	–	3.010	5.38	7.20
0.0100	1.83	3.009	–	–	3.024	5.38	7.20
0.0300	1.84	3.029	3.045	3.054	3.062	5.38	7.20
0.1000	1.93	3.143	3.160	3.186	3.192	5.38	7.24
0.2500	2.11	3.368	3.401	3.473	3.478	5.44	7.29
0.5000	2.40	3.742	3.760	3.958	3.964	5.47	7.43
0.7500	2.70	4.118	4.131	4.458	4.463	5.64	7.65
1.0000	3.01	4.490	4.511	4.969	4.972	6.01	7.90
1.5000	3.60	5.241	5.250	6.027	6.032	6.77	8.50
2.3000	4.56	6.430	6.450	7.828	7.833	7.84	9.56
4.0000	–	–	9.000[a]	12.132	12.136	–	12.136

[a] $+10^{-11}$.

determined by its variance, skew and kurtosis. In order to determine the $B_1 - B_2$ domain in which the Pearson method works correctly, one can put the variance to one. The variance is then regarded as a scaling factor. Given certain B_1 values, Table 1 provides B_2 regions in which the Pearson method can be applied. The upper bound of main type II is obtained when the inaccuracy of u_4 reaches 10%.

Point $B_1 = 0$ with $B_2 = 1.8$ represents a uniform distribution, point $B_1 = 0$ with $B_2 = 3$ a Gaussian distribution and $B_1 = 4$ with $B_2 = 9$ a shifted negative exponential distribution with zero mean. Note that a very small "disturbance" of 10^{-11} has to be added to obtain the uniform distribution, the Gaussian distribution, and the shifted negative exponential distribution to work properly. The presented method is unable to cope with these distributions if the exact moment values would be used (so without the disturbance of 10^{-11}). Each of the three distributions (uniform, Gauss, and shifted negative exponential) constitute a lower bound of Pearson type I, II, and III, respectively, for specific B_2 values. The "disturbance" of 10^{-11} results in a negligible error and shows at the same time the strength of our method to reach to the utmost end of a range. The presence of uncovered areas in Table 1 is partly due to theoretical reasons and partly to a limitation in the computing accuracy. For instance, in the gap between type I and type III, η has a very large negative value on the I side and a very large positive value on the III side.

5. Implementation of Max and Min compositions

As the implementation of the Max and Min functions are very similar, they are both discussed in this section. However, we distinguish both for Max and Min two different methods, i.e., one for n tasks with the same stochastic properties and another for two different tasks.

5.1. N-ary Max and Min composition

First, the measured four moments of the density function $\overline{f}_1(x)$ of the constituent parallel tasks are centralized according to Eq. (11), followed by the calculation of a, d_0, d_1, d_2, B_1, B_2, m, r, c, and v, while η is derived as specified by Eq. (16).

If $\eta < 0$, we apply Eq. (18):

$$\frac{1}{c_1} f_1(x) = \left[\frac{2(a_1 + x)}{a_1 + a_2}\right]^A \left[\frac{2(a_2 - x)}{a_1 + a_2}\right]^B.$$

In order to determine c_1, a summation is executed of the right-hand side of Eq. (18) of 10,000 samples over the interval $-a_1 < x < a_2$. If $0 < \eta < 1$, we apply Eq. (19):

$$\frac{1}{c_2} f_1(x) = \left[1 + \left(\frac{x}{c} - \frac{v}{r}\right)^2\right]^{-m} e^{-v \arctan\left(\frac{x}{c} - \frac{v}{r}\right)}.$$

c_2 is determined by a summation of 10,000 samples of the right hand side of Eq. (19) for the interval $-15\sqrt{u_2} < x < 15\sqrt{u_2}$. If $\eta > 1$ and positive skew, we apply Eq. (21):

$$\frac{1}{c_3} f_1(x) = \left(\frac{a_1 + x}{a_1 - a_2}\right)^A \left(\frac{2(a_2 + x)}{a_2 + 15\sqrt{u_2}}\right)^B.$$

c_3 is determined by a summation of 10,000 samples of the right-hand side of Eq. (21) for the interval $-a_2 < x < 15\sqrt{u_2}$.

The numerical integration method related to Eq. (8) for Max and Eq. (10) for Min starts by deriving the value of $f_1(x)$ at the first sample point. Within our C program this sample value of $f_1(x)$ is derived, and thereafter we step successively to the second point, third point, etc. At each point $f_1(x)$ is found and also the accumulated sum $F_1(x)$ of all the previous $f_1(x)$ samples. $f(x)$ is then found in accordance with Eq. (5) for Max and Eq. (7) for Min. The $f(x)$ of each sample point is multiplied by x, x^2, x^3, and x^4 to calculate the different moments. Again, the accumulated sum values of $xf(x)$, $x^2f(x)$, $x^3f(x)$, and $x^4f(x)$ of all previous samples are calculated (Eq. (6) for Max and Eq. (8) for Min). When the sampling terminates at the end of the valid x region, the four moments of the composite task of the centralized constituent tasks have been derived. Instead of determining $xf(x)$, $x^2f(x)$, $x^3f(x)$, and $x^4f(x)$ for every sample point, one can also calculate $(x+m)f(x)$, $(x+m)^2f(x)$, $(x+m)^3f(x)$ and $(x+m)^4f(x)$ to obtain the four moments of the composite task related to the n measured constituent tasks. The complexity of the algorithm is determined by the integration loop which involves approximately 50 floating point operations per integration step, which is totally independent of n. For the accuracy needed in our experiments, only 10,000 (Euler) integration steps are required, resulting in around 500,000 floating point operations for the computation of the four moments. This amount could even be reduced when more advanced integration methods would be used. In summary, the algorithm pairs excellent accuracy at minimum computing cost. A C implementation of the algorithm, compiled with gcc 2.95 (default optimization), only takes 0.040 CPU

s on a 750 Mhz AMD Duron processor running Linux Debian 2.2. Moreover, the execution time is totally independent of n (number of parallel tasks).

5.2. Binary Max and Min composition

In this section we determine the characteristics of the composite task consisting of two different tasks. The computer program starts by denoting the task with the smaller mean as $\overline{f}_1(x)$ and the other task as $\overline{f}_2(x)$. Both $\overline{f}_1(x)$ and $\overline{f}_2(x)$ are then shifted along the x-axis to make their means zero and are subsequently denoted $f_1(x)$ and $f_2(x)$, respectively. Next the parameters a, d_0, d_1, d_2, m, B_1, B_2, r, v, and c are calculated both for $f_1(x)$ and $f_2(x)$.

The computer program determines both for $f_1(x)$ and $f_2(x)$ which one of the three main Pearson types is applicable. To calculate the related scaling constant (either c_1, c_2, c_3, or c_4) for both $f_1(x)$ and $f_2(x)$, we have taken the bounds somewhat differently from the n-ary case. To explain this we investigate the composite density function derived by differentiation of Eq. (2) for Max and Eq. (4) for Min. This results in

$$\begin{aligned} \overline{f}(x) &= \overline{f}_1(x)\overline{F}_2(x) + \overline{f}_2(x)\overline{F}_1(x) \quad \text{for Max,} \\ \overline{f}(x) &= \overline{f}_1(x)\{1-\overline{F}_2(x)\} + \overline{f}_2(x)\{1-\overline{F}_1(x)\} \quad \text{for Min.} \end{aligned} \tag{23}$$

If we shift both $\overline{f}_1(x)$ and $\overline{f}_2(x)$ along the x-axis to zero the mean of $\overline{f}_1(x)$, $\overline{f}_2(x)$ retains a new mean which equals φ, where $\varphi = t_1 - m_1$, and m_1 and t_1 are the means of $\overline{f}_1(x)$ and $\overline{f}_2(x)$, respectively. When the mean of $\overline{f}_1(x)$ has been reduced to zero, we obtain from Eq. (23):

$$\begin{aligned} f(x) &= f_1(x)F_2(x-\varphi) + F_1(x)f_2(x-\varphi) \quad \text{for Max,} \\ f(x) &= f_1(x)\{1-F_2(x-\varphi)\} + \{1-F_1(x)\}f_2(x-\varphi) \quad \text{for Min.} \end{aligned} \tag{24}$$

Let the bounds of the centralized $f_1(x)$ be $-l_1$ and k_1, and of the centralized $f_2(x)$ be $-l_2$ and k_2. The bounds for Eq. (24) must be such that both functions $f_1(x)$ and $f_2(x-\varphi)$ must be covered. The following applies both to the Max and Min composition. Let the bounds for $f(x)$ be between $-l$ and $+k$. Then it holds that

$$\begin{aligned} l &= l_1 \quad \text{if } l_1 > l_2 - \varphi, \\ l &= l_2 - \varphi \quad \text{if } l_1 < l_2 - \varphi, \\ k &= k_1 \quad \text{if } k_1 > k_2 + \varphi, \\ k &= k_2 + \varphi \quad \text{if } k_1 < k_2 + \varphi. \end{aligned}$$

To ensure that the scaling constants (being either c_1, c_2, c_3, or c_4) of both functions remain valid in Eq. (24), one must keep the number of samples between $-l_1$ and k_1 and between $-l_2$ and k_2 used in the determination of the individual scaling constants the same as in the integration of $f(x)$. Therefore we have used the bounds $-l$ and k also for determining the scaling constants of the centralized $f_1(x)$ and the biased $f_2(x-\varphi)$.

To obtain the four moments of the composite of two parallel tasks with means m_1 and t_1, respectively, each sample of $f(x)$ is multiplied by $x + m_1$, $(x + m_1)^2$, $(x + m_1)^3$, and $(x + m_1)^4$ and the accumulating sums are obtained in order to derive P_1, P_2, P_3, and P_4, where P_1, P_2, P_3, and P_4 are the moments of the composite task.

6. Test results

The quality of our prediction method is defined by the relative errors between the moments predicted by our method and the calculated exact values or measured values in the case of well known distributions or real life programs respectively.

$$\epsilon_k = \text{error in the } k\text{-th moment} = \frac{\text{predicted value} - \text{calculated(or measured)value}}{\text{predicted value}} \times 100\%.$$

In Section 6.1 we discuss the Max and Min results of the n-ary case for some well known distributions such as the uniform, Gaussian and shifted negative exponential distribution. We have selected these three distributions as they are well-known and each represents a Pearson type I, II, and III distribution, respectively. In Section 6.2 the Max and Min results are given of the binary case for some well known distributions. In Section 6.3 we analyze the Max results of the n-ary case for some real life parallel programs.

6.1. The n-ary case for some well known distributions

6.1.1. Uniform distribution

Fig. 2 provides ϵ_k values for the Max function of a uniform distribution with zero mean and unit variance as a function of n. The uniform distribution has a B_1 value of

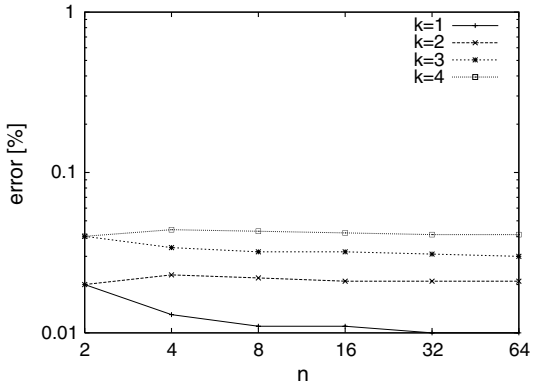

Fig. 2. ϵ_k for the Max function of a uniform distribution $\sigma = 1$.

0.0 and a B_2 value of 1.8 and corresponds to the lower limit of Pearson type I in Table 1. The calculated exact moments corresponding to Figs. 2 and 3 are:

$$\begin{aligned} E[x] &= -\sqrt{3}\left(1-\frac{2n}{n+1}\right)\sigma, \\ E[x^2] &= 3\left(1-\frac{4n}{n+1}+\frac{4n}{n+2}\right)\sigma^2, \\ E[x^3] &= 3\sqrt{3}\left[1-\frac{6n}{n+1}+\frac{12n}{n+2}-\frac{8n}{n+3}\right]\sigma^3, \\ E[x^4] &= 9\left[1-\frac{8n}{n+1}+\frac{24n}{n+2}-\frac{32n}{n+3}+\frac{16n}{n+4}\right]\sigma^4. \end{aligned} \tag{25}$$

Fig. 3 provides ϵ_k values for the Max function of a uniform distribution for different σ and $n = 2$. σ could be varied over a very wide range from 10^{-5} to at least 8 with little change in ϵ_k values. Fig. 4 provides ϵ_k values for the Min function of a uniform distribution for different n and a variance of 1. The ϵ_1 values in Figs. 4 and 5 are smaller than 0.01% and have been omitted. The calculated exact moments used in Figs. 4 and 5 are the same as in (25) except that $E[x]$ and $E[x^3]$ have now opposite signs. For the extreme situation $n = 64$ and $\sigma = 8$ the errors are still within 0.04% (Max, Min, $k = 4$).

6.1.2. Gaussian distribution

Fig. 6 provides ϵ_k values for the Max function of a Gaussian distribution with zero mean and a variance of one as a function of n. The Gaussian distribution has a B_1 value of 0.00 and a B_2 value of 3.00 and corresponds to the lower limit of Pearson type II in Table 1. Due to the impossibility to calculate the integral of a Gaussian

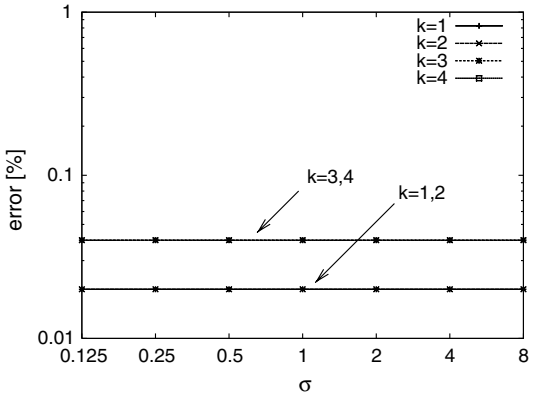

Fig. 3. ϵ_k for the Max function of a uniform distribution $n = 2$.

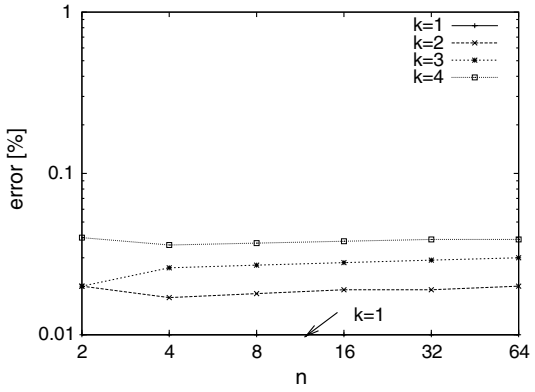

Fig. 4. ϵ_k for the Min function of a uniform distribution $\sigma = 1$.

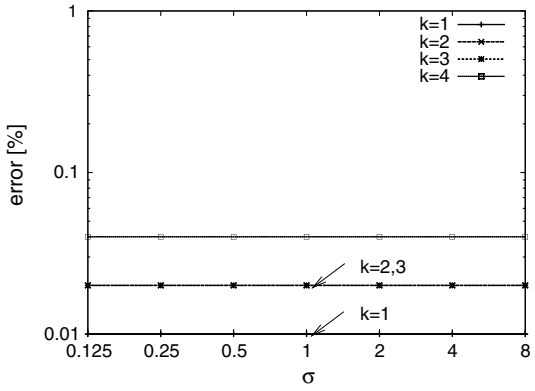

Fig. 5. ϵ_k for the Min function of a uniform distribution $n = 2$.

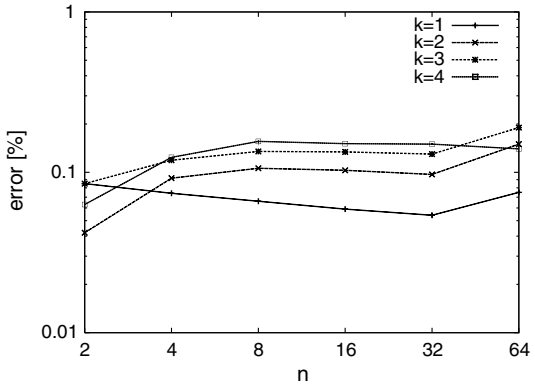

Fig. 6. ϵ_k for the Max function of a Gaussian distribution $\sigma = 1$.

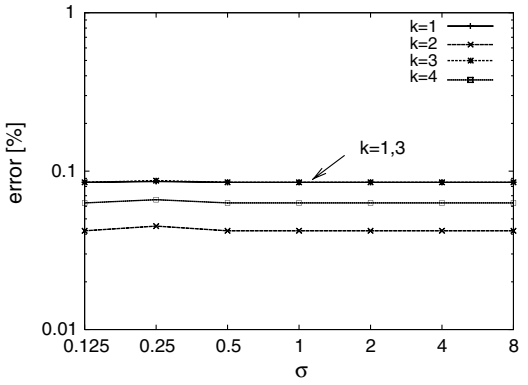

Fig. 7. ϵ_k for the Max function of a Gaussian distribution $n = 2$.

distribution we were unable to derive the exact moments corresponding to Fig. 6, but the exact moments corresponding to Fig. 7 are given below:

$$\begin{aligned} E[x] &= \frac{\sigma}{\sqrt{\pi}}, \\ E[x^2] &= \sigma^2, \\ E[x^3] &= \frac{2.5\sigma}{\sqrt{\pi}}, \\ E[x^4] &= 3\sigma^4. \end{aligned} \tag{26}$$

The values in Fig. 6 were obtained using Maple (version 9). σ could be varied over a very wide range from 10^{-5} to at least 8. Fig. 8 provides ϵ_k values for the Min function of a Gaussian distribution with zero mean and unit variance. The calculated exact moments used to obtain the numbers in Fig. 9 are the same as in Eq. (26) except that $E[x]$ and $E[x^3]$ have now opposite signs. Even in the extreme situation $n = 64$ and $\sigma = 8$ the errors are still within 4% (Max, Min, $k = 4$).

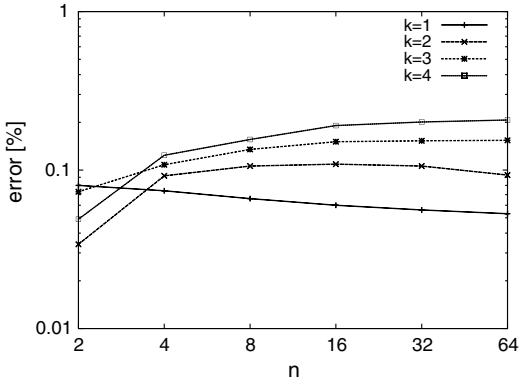

Fig. 8. ϵ_k for the Min function of a Gaussian distribution $\sigma = 1$.

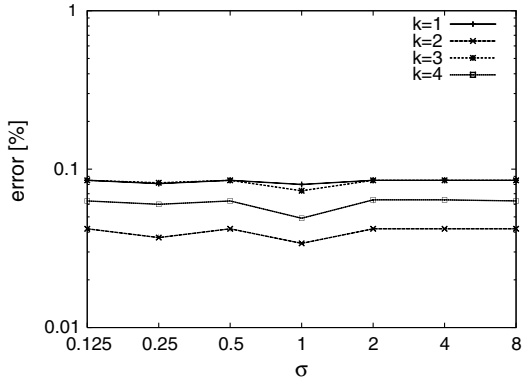

Fig. 9. ϵ_k for the Min function of a Gaussian distribution $n = 2$.

6.1.3. Shifted negative exponential distribution

The shifted negative exponential distribution has a B_1 value of 4.00 and a B_2 value of 9.00 and corresponds to the lower limit of Pearson type III in Table 1. The ϵ_k values for the Max function of a shifted negative exponential distribution with zero mean, a variance of one and n running from 2 to 64 are all below 0.01% and a plot has therefore been omitted. The same applies to the distribution with n is two and σ varying between 0.125 and 8 and a plot has been left out for the same reason. Even in the extreme situation $n = 64$ and $\sigma = 8$ the errors are still within 4% (Max, $k = 4$). σ could be varied over a large range from 10^{-2} to at least 8. Fig. 10 provides ϵ_k values for the Min function of a shifted negative exponential distribution with zero mean and unit variance. The exactly calculated moments corresponding to Figs. 10 and 11 are:

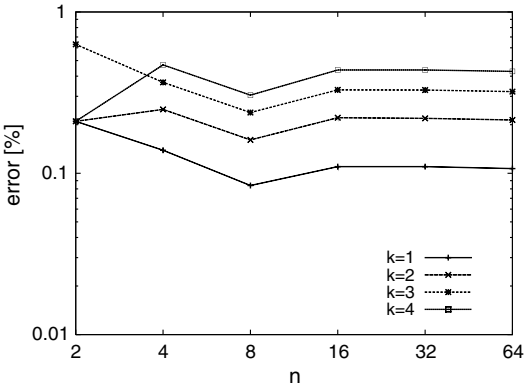

Fig. 10. ϵ_k for the Min function of a shifted negative exponential distribution $\sigma = 1$.

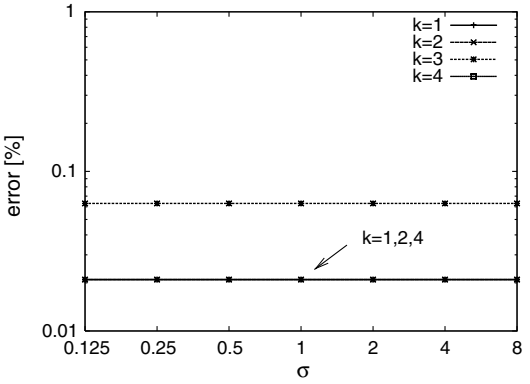

Fig. 11. ϵ_k for the Min function of a shifted negative exponential distribution $n = 2$.

$$
\begin{aligned}
E[x] &= \left(\frac{1}{n} - 1\right)\sigma, \\
E[x^2] &= \left(\frac{2}{n^2} - \frac{2}{n} + 1\right)\sigma^2, \\
E[x^3] &= \left(\frac{6}{n^3} - \frac{6}{n^2} + \frac{3}{n} - 1\right)\sigma^3, \\
E[x^4] &= \left(\frac{24}{n^4} - \frac{24}{n^3} + \frac{12}{n^2} - \frac{4}{n} + 1\right)\sigma^4.
\end{aligned}
\tag{27}
$$

Fig. 11 provides ϵ_k values for the Min function of a shifted negative exponential distribution for different σ and $n = 2$. σ could be varied over a large range from 10^{-2} to at least 8. Even in the extreme situation $n = 64$ and $\sigma = 8$ the errors are still within 0.5% (Min, $k = 4$).

6.2. The binary case for some well-known distributions

The quality of our prediction method is given in this section when applied to sets of two different tasks. Three combinations have been investigated: uniform-Gaussian, uniform-shifted negative exponential, and Gaussian-shifted negative exponential. The density functions of the uniform, shifted negative exponential, and Gaussian distributions had zero means. The experiments have been carried out over a wide range of standard deviations, for the uniform and Gaussian distributions from $\sigma = 10^{-5}$ to at least 8, and for the shifted negative exponential distribution from $\sigma = 10^{-2}$ to at least 8. We report here only the results for combinations where each of the two distributions has $\sigma = 1$.

6.2.1. The Max function

The results of the Max function of the combination uniform-Gauss showed $\epsilon_1 = 0.020\%$, $\epsilon_2 = 0.007\%$, $\epsilon_3 = 0.017\%$, and $\epsilon_4 = 0.013\%$. For the combination

uniform-shifted negative exponential, these four ϵ values are 0.003%, 0.003%, 0.005%, 0.008%, respectively, and for the combination Gauss-shifted negative exponential 0.086%, 0.035%, 0.082%, and 0.185%, respectively.

6.2.2. The Min function

The results of the Min function of the combination uniform-Gauss showed $\epsilon_1 = 0.041\%$, $\epsilon_2 = 0.025\%$, $\epsilon_3 = 0.018\%$, and $\epsilon_4 = 0.006\%$. For the combination uniform-shifted negative exponential these four ϵ values are 0.010%, 0.020%, 0.038%, and 0.077%, respectively, and for the combination Gauss-shifted negative exponential 0.062%, 0.036%, 0.074%, and 0.302%, respectively.

6.3. The n-ary case for real-life programs using the Max procedure

Results are given of the completion time accuracies of a set of n parallel programs, all having the same stochastic properties. The accuracies are obtained against the use of simulation results. The errors in the four moments of the composite task have now been denoted δ_1, δ_2, δ_3, and δ_4 in order to indicate the difference with the ϵ values which were based upon zero means of the constituent tasks. Originally we used 6000 simulation runs per case [20] but, in view of the excellent accuracy of our current method, this number was found to be too low and has been increased to 100,000 in this paper. The reason stems from the measurement inaccuracy of the mean termination time of a constituent task as well as the measurement inaccuracy of the mean completion time of its resulting composite task. These errors contribute to the δ_1 error between the predicted and measured mean completion time of the composite (something similar applies to the δ_2, δ_3, and δ_4 values). The standard variance of a constituent task and of the composite task were added and denoted σ^2. Under the condition that the means of a constituent task and the composite task do not differ too much, the measurement accuracy with which δ_1 values can be determined is then limited to

$$\frac{1.96\sigma}{\sqrt{\#\text{ samples}}\cdot \text{mean of composite}} \times 100\%$$

with 95% confidence. We found that in nearly all cases in Section 6.3, the 95% confidence mean values were greater than the δ_1 values when using 6000 simulation runs. For that reason the tables below are based upon 100,000 simulation runs providing 95% confidence accuracies of approximately the same values as the measured δ_1 values. The errors caused by the algorithm are therefore sometimes smaller than the corresponding δ values given in Tables 3 and 5 below. To back up this conclusion further we also executed the three experiments of Table 2 with $n = 1$, thus with only

Table 2
Measured and calculated execution time parameters of WATOR constituent tasks

i	n	$\mathsf{E}(x)$	$V(x)$	$S(x)$	$K(x)$	η	B_1	B_2
4	4	489.52	1742.80	0.0699	2.976	−0.0098	0.00489	2.9758
4	16	489.79	1753.28	0.0064	3.036	+0.0004	0.00004	3.0358
4	64	489.68	1703.30	0.0685	2.960	−0.037	0.00469	2.9599

one task. The generated output must then be equal to the input moments, which was found to be true, up to an accuracy of 10^{-4} or even 10^{-5}.

To verify the accuracies of the Max procedure in the context of real-life programs we used a simulation program WATOR [9] and a program called PSRS (parallel sorting by regular sampling) [23]. The choice of programs is based on the fact that both are characteristic of two large classes of programs that exhibit stochastic execution time behavior, i.e., simulations that typically involve intentional random behavior, and programs that have an execution time that is largely data-dependent, respectively. The programs are run 100,000 times with random input configurations (while maintaining constant problem size parameters such as grid size for WATOR and vector size for PSRS). From these runs the overall execution time distribution is measured in terms of the first four moments, as well as the moments of the constituent task execution time. The task measurements are used as the basis for the prediction scheme based on the Pearson method, the results of which are compared to the measured moments of the overall execution time, mentioned before.

6.3.1. WATOR

WATOR is a Monte Carlo simulation in which virtual fish and sharks live, move randomly, breed, and eat one another in a two-dimensional ocean [3]. In our experiment we perform a simple rectangular subdomain decomposition of the ocean such that each processor is assigned to process 2500 grid points. On initialization, the number of fish within each processor is randomly generated. The four parameters of the execution time parameters of the constituent tasks in Table 2 are designated by the mean $\mathsf{E}(x)$, variance $V(x)$, skew $S(x)$, and kurtosis $K(x)$. The number of parallel tasks is n. An iteration i defines the workload of a task to move, breed and eat. For i is zero, such a task is executed only once. When i increases a clustering and correlation of data begin to appear. This correlation is still very small when i is four. The errors of the four moments of the composite execution time and the number of parallelism as depicted in Table 2 are presented in Table 3. The parameters η, B_1, and B_2 are given in Table 2 as well. Note that the Pearson main types I and II have been used.

6.3.2. PSRS

The PSRS algorithm (parallel sorting by regular sampling) comprises three sections in series with different levels of correlation between the parallel tasks in each section. The first section (SORT) divides an array in n equal sub-arrays. All sub-

Table 3
Errors of the four moments of the composite execution time for WATOR when the constituent n tasks have parameters as depicted in Table 2

i	n	δ_1 (%)	δ_2 (%)	δ_3 (%)	δ_4 (%)
4	4	+0.212	+0.418	+0.618	+0.811
4	16	+0.184	+0.368	+0.541	+0.716
4	64	−0.043	−0.098	−0.165	−0.245

Table 4
Measured and calculated execution time parameters for SORT and DISJOIN constituent tasks

Section	n	$\mathsf{E}[x]$	$V[x]$	$S[x]$	$K[x]$	η	B_1	B_2
SORT	4	9589	108,272	0.8944	4.267	+5.388	0.7992	4.267
	16	9590	113,265	0.9817	4.379	−6.706	0.9638	4.379
	64	9592	113,692	1.0694	4.906	+2.892	1.1437	4.906
DISJOIN	4	751.6	157.83	−0.3696	4.143	+0.057	0.1366	4.143
	16	960.8	50.25	−0.1669	3.207	+0.064	0.0279	3.207
	64	1056.3	7.60	−0.4104	3.336	+0.787	0.1684	3.336

Table 5
Errors of the four moments of the composite execution time for SORT and DISJOIN when constituent tasks have parameters as in Table 4

Section	n	δ_1 (%)	δ_2 (%)	δ_3 (%)	δ_4 (%)
SORT	4	−0.056	−0.119	−0.188	−0.256
	16	+0.022	+0.042	+0.062	+0.079
	64	+0.232	+0.463	+0.693	+0.923
DISJOIN	4	−0.094	−0.185	−0.272	−0.355
	16	+0.070	+0.144	+0.223	+0.306
	64	+0.009	+0.018	+0.028	+0.037

arrays are sorted in parallel, after which a number of global pivots are determined. Based upon these pivots, the second section (DISJOIN) redivides the sub-array in n parts. The third section (MERGE) merges the sub-arrays cyclically with regard to the processor index. As a result, a sorted array is obtained. A description of the algorithm can be found in [23]. We present the empirical results of the first two sections as the third section (MERGE) shows too large a correlation. Table 4 depicts the execution time parameters of constituent tasks of SORT and DISJOIN when n is 4, 16, and 64. The errors of the four moments of the composite execution time with the workloads and the amount of parallelism as given in Table 4 are presented in Table 5. All main Pearson types I, II, and III have been used.

7. Conclusion

Two methods are presented to predict the execution time of a combination of parallel tasks which have a stochastic workload. The presented methods are especially of interest when handling parallel, independent, time-critical applications for which other performance prediction methods are usually inadequate. The independence of parallel processing tasks means that the methods are most appropriate for tasks with a minimum of resource sharing. The two mathematical methods are the Max procedure and the Min procedure. The Max procedure is applied in the general case that the composite processing task completes at the time its longest constituent task terminates. However, there are also examples where the completion of its shortest

processing task terminates the whole parallel processing job such as in seach-programs. In the latter case we have applied the Min procedure. The constituent tasks can be n stochastically equal parallel programs (n-ary case) or two different programs (binary case). The method is based on the Pearson characterization of the density function of a constituent task of which the first four moments are known. The Pearson equations have been slightly modified to avoid numeric overflow. Furthermore, the skew^2-kurtosis domain ($B_1 - B_2$) has been mapped in which the main Pearson types can be used safely. The predicted values of both n-ary and binary compositions, using Max and Min procedures have been compared with calculated exact results of functions having a uniform, Gaussian, or shifted negative exponential density distribution for a wide range of standard deviations and number of parallel tasks. The algorithm has a very low complexity of around 500,000 floating point operations, that is totally independent of n (number of parallel tasks). The results show excellent accuracy of better than 1% with a very few exceptions in extreme situations (the error would increase to 4% in that case). In addition, the execution times of a number of real-life programs have been investigated in the n-ary case with a parallelism of 4, 16 and 64. The errors in the nine n-ary experiments shown in Tables 3 and 5 are below 1%. A C implementation of the algorithm, compiled with gcc 2.95 (default optimization), only takes 0.040 CPU s on a 750 MHz AMD Duron processor running Linux Debian 2.2.

Acknowledgements

The authors gratefully acknowledge the help of Hasyim Gautama in providing the measurement data on the benchmark program experiments. The authors also wish to extend their gratitude to the referees for their comments which enabled us to significantly improve the paper.

References

[1] F. Allen et al., A framework for determining useful parallelism, in: Proc. 1988 Int. Conf. Parallel Proc., IEEE, 1988, pp. 207–215.
[2] D. Ataputtu, D. Gannon, Building analytical models into an interactive prediction tool, in: Proc. Supercomputing'89, ACM, 1989, pp. 521–530.
[3] I. Angus et al. Solving Problems on Concurrent Processors, Software for Concurrent Processors, vol. 2, Prentice Hall, 1990.
[4] V. Balasundaram et al., A static performance estimator to guide data partitioning decisions, in: Proc. 3rd AMC SIGPLAN Symposium on PPoPP, April 91.
[5] M.J. Clement, M.J. Quinn, Multivariate statistical techniques for parallel performance prediction, in: Proc. 28th Hawaii Int. Conf. on System Sciences, vol. 2, IEEE, 1995, pp. 446–455.
[6] W.P. Elderton, N.L. Johnson, Systems of Frequency Curves, Cambridge University Press, 1969, pp. 35–81.
[7] T. Fahringer, H.P. Zima, A static parameter-based performance prediction tool for parallel programs, in: Proc. 7th ACM Int. Conf. on Supercomputing, Tokyo, July 1993, pp. 207–219.
[8] H. Gautama, A.J.C. van Gemund, Symbolic performance prediction of speculative parallel programs, Parallel Processing Letters 13 (4) (2003) 513–524.

[9] H. Gautama, A Statistical Approach to the Performance Modeling of Parallel Systems, PhD Thesis, Delft University of Technology, The Netherlands, 2004.
[10] H. Gautama, A.J.C. van Gemund, Performance prediction of stochastic parallel task compositions, IEEE Transactions in Parallel and Distributed Systems, in press.
[11] E. Gelenbe, E. Montagne, R. Suros, A performance model of block structured parallel programs, in: M. Cosnard et al. (Eds.), Parallel Algorithms and Architectures, Elsevier Science Publishers, North-Holland, 1986, pp. 127–138.
[12] E.J. Gumbel, Statistical theory of extreme values (main results), in: A.E. Sarhan, B.G. Greenberg (Eds.), Contributions to Order Statistics, John Wiley & Sons, 1962, pp. 56–93.
[13] C.P. Kruskal, A. Weiss, Allocating independent subtasks on parallel processors, in: Proc. 1986 Int. Conf. Parallel Proc., IEEE, 1986, pp. 145–152.
[14] B.P. Lester, A system for computing the speedup of parallel programs, in: Proc. 1986 Int. Conf. Parallel Proc., IEEE, 1986, pp. 145–152.
[15] D.-R. Liang, S.K. Tripathi, On performance prediction of parallel computations with precedence constraints, IEEE Transactions on Parallel and Distributed Systems 11 (5) (2000) 491–508.
[16] S. Madala, J. Sinclair, Performance of synchronous parallel algorithms with regular structures, IEEE Transactions on Parallel and Distributed Systems 2 (1) (1991) 105–116.
[17] C.L. Mendes, D.A. Reed, Integrated compilation and scalability analysis for parallel systems, in: Proc. Int. Conf. on Parallel Architectures and Compilation Techniques (PACT'98), Paris, October 1998, pp. 385–392.
[18] C.D. Polychronopoulos et al., Parafrase-2: an environment for parallelizing partitioning, synchronizing and scheduling programs on multiprocessors, in: Proc. 1989 Int. Conf. Parallel Proc., IEEE, 1989, pp. 2:39–2:48.
[19] J.S. Ramberg et al., A probability distribution and its uses in fitting data, Technometrics 21 (2) (1979) 201–214.
[20] G.L. Reijns, A.J.C. Van Gemund, H. Gautama, On the use of pearson distributions for the performance prediction of parallel programs, in: Proc. Symposium on Performance Evaluation—Stories and Perspectives, Vienna, 5–6 December 2003, ISBN 3-85403-175-0, Oesterreichische Computer Gesellschaft.
[21] R.A. Sahner, K.S. Trivedi, Performance and reliability analysis using directed acyclic graphs, IEEE Transactions on Software Engineering 13 (10) (1987) 1105–1114.
[22] V. Sarkar, Partitioning and Scheduling Parallel Programs for Multi-processors, Pitman, 1989.
[23] H. Shi, J. Schaeffer, Parallel sorting by regular sampling, Journal of Parallel and Distributed Computing 14 (4) (1992) 361–372.
[24] K. So et al., A speedup analyzer for parallel programs, in: Proc. 1987 Int. Conf. Parallel Proc., IEEE, 1987, pp. 653–661.
[25] A. Stuart, J.K. Ord, Kendall's Advanced Theory of Statistics, vol. 1, 6th ed., Halsted Press, New York, 1994.
[26] K.S. Trivedi, Probability and Statistics with Reliability, Queuing and Computer Science Applications, Prentice Hall, 1982.