Symbolic Performance Prediction of Speculative Parallel Programs

Hasyim Gautama and Arjan J.C. van Gemund

Delft University of Technology
P.O. Box 5031, NL-2600 GA Delft, The Netherlands
{h.gautama, a.j.c.vangemund}@its.tudelft.nl

Abstract. Speculative parallelism refers to searching in parallel for a solution, such as finding a pattern in a data base, where finding the first solution terminates the whole parallel process. Different performance prediction methods are required as compared to traditional parallelism. In this paper we introduce an analytical approach to predict the execution time distribution of data-dependent parallel programs that feature N-ary and binary speculative parallel compositions. The method is based on the use of statistical moments which allows program execution time distribution to be approximated at $O(1)$ solution complexity. Measurement results for synthetic distributions indicate an accuracy that lies in the percent range while for empirical distributions on internet search engines the prediction accuracy is acceptable, provided sufficient workload unimodality.

1 Introduction

Analytically predicting the execution time distribution of dependent parallel programs is an extremely challenging problem. Even for fixed problem size the variety of typical input data sets may cause a considerable execution time variance. Especially for time-critical applications, merely predicting the mean execution time does not suffice and knowledge on the execution time distribution is essential. In particular, the execution time distribution of a *parallel* composition of tasks with stochastic workloads is computationally demanding.

Parallel composition can be distinguished in *and*-parallel and *or*-parallel compositions. Of the two forms, and-parallel composition is most common in parallel computing, and typically results from *task* or *data* parallelism, where each task essentially involves different computation (workload), or the same computation on different data, respectively. Consider an and-parallel composition of N tasks having an execution time X_i. As in and-parallelism the slowest computation (largest workload) determines overall execution time Y, it follows that

$$Y = \max_{i=1}^{N} X_i. \tag{1}$$

Or-parallel composition, in contrast, results from *speculative* parallelism where each task is initiated without being sure that its execution should complete. Instead of and-parallel composition, an or-parallel composition terminates

H. Kosch, L. Böszörményi, H. Hellwagner (Eds.): Euro-Par 2003, LNCS 2790, pp. 88–97, 2003.

when *one* of the parallel tasks has been completed. Examples of or-parallel composition include searching in parallel for a solution, such as finding patterns in a data base, where the first solution found terminates the whole parallel process. As the fastest task completion now determines the performance, the overall execution time Y is given by

$$Y = \min_{i=1}^{N} X_i. \tag{2}$$

Other areas where computing the minimum of a set of stochastic numbers is of interest include reliability analysis (e.g., system breakdown is determined by earliest component failure), and road traffic analysis (e.g., speed of a vehicle chain on a single lane is determined by the slowest).

In this paper we consider the solution of Eq. (2) which has received less attention compared to Eq. (1), despite its obvious utility. While many authors have studied Eq. (1) as part of their prediction techniques [2,3,5,6,7,8,9,10,13,14, 15,16], some authors also include Eq. (2) in their analysis [6,9,14], but restrict the analysis either to specific distributions rarely found in (parallel) programs, or assume distributions to be symmetric [6].

When X_i are deterministic Eq. (2) provides an exact and low-cost prediction tool. When, as in data-dependent programs, X_i is stochastic, interpreting Eq. (2), e.g., in terms of mean values in the sense of

$$\mathsf{E}[Y] = \min_{i=1}^{N} \mathsf{E}[X_i] \tag{3}$$

produces a severe error, which for symmetric distributions X_i is linear in the variance of X_i and logarithmic in N [6]. The same applies to Eq. (1).

Solving Eq. (2) presents a major problem that is well-known in the field of order statistics. There are a number of approaches to express an execution time distribution, the choice of which largely determines the trade-off between accuracy and cost involved in solving Y. An exact, closed-form, solution for the distribution of Y is expressed by using the cumulative density function (cdf) [18]. Unfortunately, many execution time distributions do not have a closed-form cdf function, such as the normal distribution and most of the distributions obtained from measurements. Alternative approaches restrict the type of distribution to those for which Eq. (2) can be solved, such as the class of exponomial distributions [14] or Erlang and Hyperexponential distributions [9]. While proven a powerful tool for, e.g., reliability modeling, such a distribution workload rarely occurs in practice.

Another, more general way of expressing distributions is based on approximating an execution time distribution in terms of a limited number of statistical moments. In [1] an analytic performance prediction technique based on the first four statistical moments is presented for sequential and conditional task compositions. The raw moment of X, denoted $\mathsf{E}[X^r]$, is defined by

$$\mathsf{E}[X^r] = \int_{-\infty}^{\infty} x^r \mathrm{d}F_X(x). \tag{4}$$

Although the approach is straightforward in the sequential domain, for parallel composition, however, there is in general no analytic, closed-form solution for $\mathsf{E}[Y^r]$ in terms of $\mathsf{E}[X^r]$ due to the following integration problem. From Eqs. (2) and (4) we obtain

$$\mathsf{E}[Y^r] = \int_{-\infty}^{\infty} x^r \mathrm{d}(1 - (1 - F_X(x))^N). \tag{5}$$

The right-hand side integration is fundamentally impossible to solve analytically in such a way that $\mathsf{E}[Y^r]$ can be expressed directly as function of $\mathsf{E}[X^r]$. This problem becomes even more complicated when different workloads are involved.

Recently, a similar integration problem, arising from the *and*-parallel case, has been solved, by introducing generalized lambda distributions as an approximation of the execution time distribution [2,3]. Based on a similar approach, in this paper we solve the integration problem in Eq. (5) for the *or*-parallel composition, for both N-ary or-parallel composition (identical workload distributions), and binary or-parallel composition (different workload distributions). Our contribution has the specific advantage that the approximation of $\mathsf{E}[Y^r]$, now readily expressed in terms of the first four input moments $\mathsf{E}[X^r]$, has high accuracy and $O(1)$ solution complexity. Furthermore, this approach extends the use of statistical moments, shown to be successful in the sequential domain [1] and the and-parallel domain [2,3] into the or-parallel domain. To the best of our knowledge such an approach towards solving $\mathsf{E}[Y^r]$ in terms of $\mathsf{E}[X^r]$ has not been described elsewhere.

The remainder of the paper is organized as follows. Section 2 presents the rationale behind our choice of generalized lambda distributions and describes how our closed-form solution for $\mathsf{E}[Y^r]$ is derived. The accuracy of our method is described in Section 3 using standard distributions that are characteristic of workload distributions found in practice. In Section 5 we state our conclusion.

2 Analysis

2.1 Rationale

As mentioned in Section 1, our performance prediction method is based on the use of a limited number of statistical moments (typically the first four moments, i.e., mean, variance, skewness and kurtosis) aimed to pair good accuracy with minimum solution complexity. Previously, the method has been applied to compute the execution time distribution of sequential programs based on the distribution parameters of the constituent program parts such as loops, branches, and basic blocks [1]. The method provides a very good prediction of the execution time of sequential programs, while the analysis is straightforward. In contrast, the analysis of determining the distribution of a *parallel* composition in terms of input distributions is problematic as described in Section 1. Directly applying the moment method in the analysis is fundamentally impossible due to integration problems. In order to extend the application of our moment method to parallel

composition we introduce the use of generalized lambda distributions (GLD) for two reasons. First and most importantly, due to the specific formulation of the GLD we can solve our integration problem in Eq. (5) so that it becomes possible to obtain the moments of Y. Second, unlike other approximating distributions such as, e.g., Pearson distributions, the GLD comprises of only four parameters which makes it directly compatible to our four-moment method used in the sequential domain.

2.2 Generalized Lambda Distributions

The generalized lambda distribution, $\mathrm{GLD}(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$, is a four-parameter distribution defined by the percentile function R as function of the cdf, $0 \leq F \leq 1$, according to [12]

$$X = R_X(F) = \lambda_1 + \frac{F^{\lambda_3} - (1-F)^{\lambda_4}}{\lambda_2} \tag{6}$$

where λ_1 is a location parameter, λ_2 is a scale parameter and λ_3 and λ_4 are shape parameters. While the cdf does not exist in simple closed form, the probability density function (pdf) is given by

$$f(x) = \frac{\mathrm{d}F}{\mathrm{d}R_X(F)} = \frac{1}{R'_X(F)} = \frac{\lambda_2}{\lambda_3 F^{\lambda_3 - 1} + \lambda_4 (1-F)^{\lambda_4 - 1}}. \tag{7}$$

This four-parameter distribution includes a wide range of curve shapes. Specifically, the distribution can provide good approximations to well-known densities as found in [12]. Provided that the lambda values are given, obtaining the statistical moments proceeds as follows. Without loss of generality let $\lambda_1 = 0$. Then from Eqs. (4) and (6) the rth raw moment of X is given by

$$\mathsf{E}[X^r] = \frac{1}{\lambda_2^r} \sum_{i=0}^{r} \binom{r}{i} (-1)^i B(\lambda_3(r-i)+1, \lambda_4 i + 1) \tag{8}$$

where B denotes the beta function as defined by

$$B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}\mathrm{d}t. \tag{9}$$

Since the beta function has been provided in the current standard mathematical library, the computation cost of Eq. (8) for r up to four is negligible.

To obtain the four lambda values from the moments we can refer to a table provided in [12]. If more precise lambda values are required, an alternative to looking up the table is to apply a method based on function minimization such as the Nelder-Mead simplex procedure [11]. This procedure requires a constant number of iteration steps. Even for a 10^{-10} precision for the lambda values, our experiments show that no more than 100 iteration steps are required which involves evaluating Eq. (8) (i.e., in total less than 10^5 floating point operations). Thus providing the first four moments, the cost of obtaining the lambda values from the table is also negligible.

2.3 Or-Parallel Composition

In this section we present how the GLD is applied in the analysis of speculative, or-parallel composition. Our results are stated in terms of the following two theorems for N-ary and binary compositions, respectively.

Theorem 1. The nth order statistic

Let $Y_1 \leq Y_2 \leq \ldots \leq Y_N$ be random variables obtained by permuting N independent and identical distributed (iid) variates of continuous random variables X, i.e., $X_1, X_2, \ldots, X_N$, in increasing order. Let $\mathsf{E}[X^r]$, $r = 1, 2, 3, 4$ exists while X can be expressed in terms of GLD$(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ where λ_j are functions of $\mathsf{E}[X^r]$. Without loss of generality let $\lambda_1 = 0$. Then for $n = 1, 2, \ldots, N$ the rth raw moment of Y_n is given by

$$\mathsf{E}[Y_n^r] = \frac{n}{\lambda_2^r}\binom{N}{n}\sum_{i=0}^{r}\binom{r}{i}(-1)^i B(\lambda_3(r-i)+n, \lambda_4 i + N - n + 1) \tag{10}$$

where B denotes the beta function.

□

For the proof we refer to [4].

Theorem 1 enables us to express $\mathsf{E}[Y^r]$ in terms of $\mathsf{E}[X^r]$, using the GLD by simply looking up a table [12] and subsequently using Eq. (10). Note, that our analysis technique is parametric in the problem size N (the number of tasks involved in the parallel section). Moreover, note that the solution complexity is entirely *independent* of N (i.e., $O(1)$). If more precise lambda values are required, an alternative to looking up the table is to apply a method based on function minimization such as the Nelder-Mead simplex procedure [11]. This procedure requires a constant number of iteration steps. Even for a 10^{-10} precision for the lambda values, our experiments show that no more than 100 iteration steps are required which involves evaluating Eq. (8) (i.e., in total less than 10^5 floating point operations).

As in speculative parallel composition Y is determined by the smallest X_i, the specific instance $n = 1$ of Theorem 1 applies (i.e., the smallest order statistic). Our result is stated in the following corollary.

Corollary 1. N-ary or-parallel composition

Under the same assumption as in Theorem 1 let random variable Y be defined as

$$Y = \min(X_1, X_2, \ldots, X_N). \tag{11}$$

Then the rth raw moment of Y is given by

$$\mathsf{E}[Y^r] = \frac{N}{\lambda_2^r}\sum_{i=0}^{r}\binom{r}{i}(-1)^i B(\lambda_3(r-i)+1, \lambda_4 i + N). \tag{12}$$

The proof can be straightforward obtained from Eq. (10) by substituting $n = 1$. Similar to sequential compositions the solution complexity of Eq. (12) is entirely *independent* of N (i.e., $O(1)$) while its computation cost is negligible small (cf. Eq. (8)).

While Theorem 1 and associated corollary restrict the workload X_i to be iid, for *different* workload in binary or-parallel composition we present the following theorem.

Theorem 2. Binary or-parallel composition
Let random variable Y be defined as

$$Y = \min(X_1, X_2) \tag{13}$$

where X_i are independent random variables for which $\mathsf{E}[X_i^r]$ for $r = 1, 2, 3, 4$ exist. Let X_i can be expressed in terms of $\mathrm{GLD}(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ where $\lambda_{i,j}$ are functions of $\mathsf{E}[X_i^r]$. Then the raw moments of Y are given by

$$\mathsf{E}[Y^r] = \int_0^1 (R_{X_1}(F))^r(1 - F_{X_2}(R_{X_1}(F))) + (R_{X_2}(F))^r(1 - F_{X_1}(R_{X_2}(F)))\mathrm{d}F. \tag{14}$$

□

Again, for the proof we refer to [4].

A straightforward implementation of Eq. (14) is as follows

$$\mathsf{E}[Y^r] = \frac{1}{K}\sum_{k=1}^{K}((R_{X_1}(\frac{k}{K}))^r(1-F_{X_2}(R_{X_1}(\frac{k}{K})))+(R_{X_2}(\frac{k}{K}))^r(1-F_{X_1}(R_{X_2}(\frac{k}{K})))) \tag{15}$$

where K is large. In practice, $K = 10^4$ is sufficiently large.

Based on the GLD an alternative approach to compute $F_{X_i}(R_{X_j}(k/K))$ is using Newton's method according to the following.

$$F_{X_{i,l+1}}(R_{X_j}(k/K)) = F_{X_{j,l}} - R_{X_j}(F_{X_{j,l}})/R'_{X_j}(F_{X_{j,l}}) \tag{16}$$

where $F_{X_{j,0}} = 1/K$. The use of Newton's method is appropriate since we need the GLD only to represent the execution distribution in the integration (Eq. (14)). Furthermore Newton's method converges quadratically and converges to a solution within a few iteration due to specific property of cdf. The evaluation of Eq. (15) requires 60 ms on a 1 GHz Pentium III processor.

3 Synthetic Distributions

This section describes the quality of our prediction approach when applied to some of the frequently-used standard distributions. The estimation quality is defined by the relative error ε_r evaluated from the predicted moments $\mathsf{E}[Y_p^r]$ and the measured moments $\mathsf{E}[Y_m^r]$, according to

$$\varepsilon_r = \frac{|\mathsf{E}[Y_m^r] - \mathsf{E}[Y_p^r]|}{\mathsf{E}[Y_m^r]} \tag{17}$$

where $\mathsf{E}[Y_p^r]$ may be Eq. (12) or (15) for N-ary or binary parallel composition, respectively. In all experiments in this paper we use ε_r in Eq. (17) to determine

the estimation error of our method using 6,000 data samples. For the standard distributions $\mathsf{E}[Y_m^r]$ is computed using Eq. (5) since the cdf's are known.

For the uniform distribution, $\mathsf{E}[Y^r]$ is determined exactly since the GLD and Newton's method express this distribution exactly.

The exponential distribution (used by many authors) is typical for workloads occurring when, e.g., searching for a key in a data base. The results for $\mathsf{E}[X] = 1$ ε_r [%] for N-ary parallel composition is shown by Figure 1 (left) while for binary composition where $\mathsf{E}[X_1] = 1$, and $\mathsf{E}[X_2] = 1/\theta$ ε_r [%] is shown by Figure 1 (right). The figure shows that ε_r is in the percent range and insensitive to θ value variation.

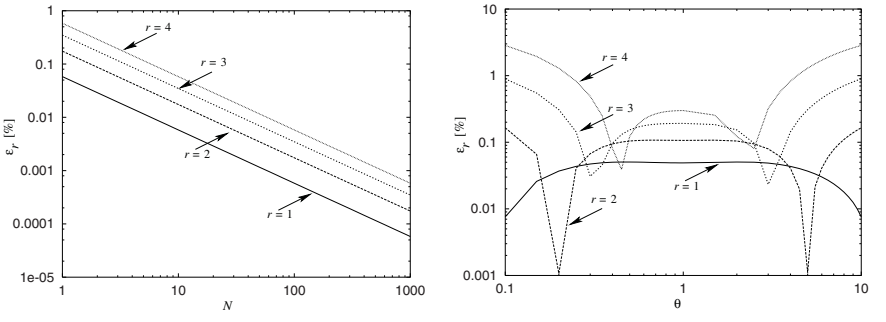

Fig. 1. ε_r [%] for N-ary and binary exponential distribution

Next, we consider the normal distribution, which is also appropriate to model workloads as found in many programs. For N-ary composition we let $\mathsf{E}[X] = 0$ and $\mathsf{Var}[X] = 1$ while N varies to 10,000. ε_r is in the percent range as shown by Figure 2. For binary composition we let $\mathsf{E}[X_1] = 1$ and $\mathsf{Var}[X_1] = 1$, while $\mathsf{E}[X_2] = \mu$ and $\mathsf{Var}[X_2] = \sigma^2$. We vary μ and σ as shown in Figure 3. In Figure 3 (left), ε_r is in the percent range and increases logarithmically as function of σ. Numerical instabilities in the computation of $\mathsf{E}[Y_m^r]$ prohibit larger values for σ. In Figure 3 (right), ε_r decreases logarithmically as function of μ, and $\varepsilon_r < 1\%$.

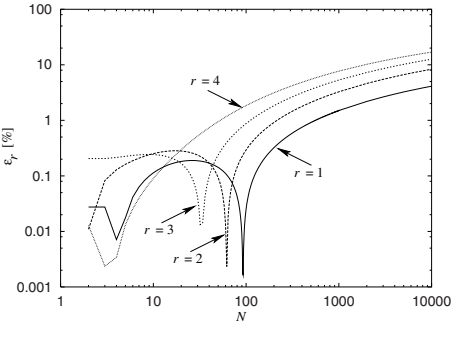

Fig. 2. ε_r [%] for N-ary normal distribution

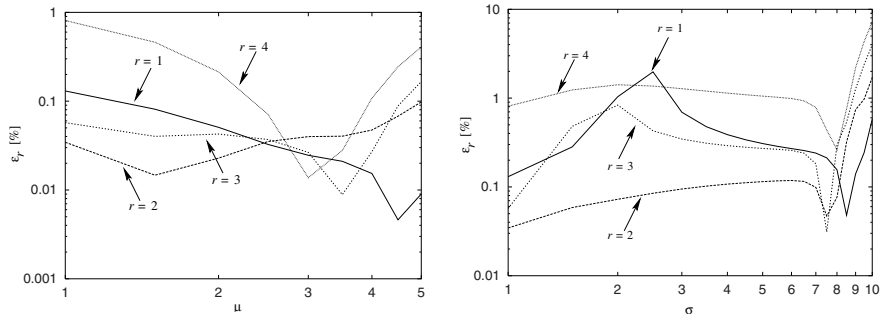

Fig. 3. ε_r [%] for binary normal distribution

4 Empirical Distributions

While the previous distributions already provide an indication of the performance of our technique, in this section we present the results for empirical distributions as measured from a distributed search application (using search engines) on the internet. For a large number of queries, each query is sent to each site in parallel, the response time being determined by the fastest response.

The search engines are located at URL addresses in the USA (dot com), Europe (dot nl), and Latin America (dot cl). As query we search 2,000 large cities around the world while as response we obtain html text, which includes the search time (in seconds) excluding internet communication delay. The workload distribution is shown in Figure 4 (left). The dot cl domain has a left-skewed distribution while the dot com and dot nl domains have bimodal distributions.

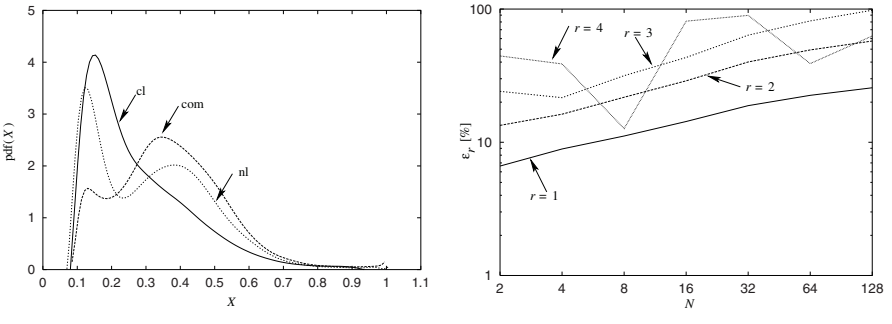

Fig. 4. Empirical distributions (left) and ε_r [%] for .cl (right)

In an N-ary or-parallel search, we represent all sites on one particular domain in terms of one workload distribution X and apply Eq. (12). To resemble identical distributions for X we repeat the search on one domain up to 128 times in unconsecutive manner to avoid probable caching mechanism in the search engine.

The prediction error is shown in Figures 4 (right) and 5 for the dot cl, dot com and dot nl domains, respectively. For all three cases ε_r increases for larger N. The average error for the dot cl domain is appreciable since the distribution is sharply left-skewed where the distribution mass is concentrated near the minimum value. In such case the fitting for lambda values is very sensitive. The error for the dot com and dot nl domains are much worse and increase rapidly for larger N, which is caused by the fact that the distribution of X is no longer unimodal, a requirement of our technique.

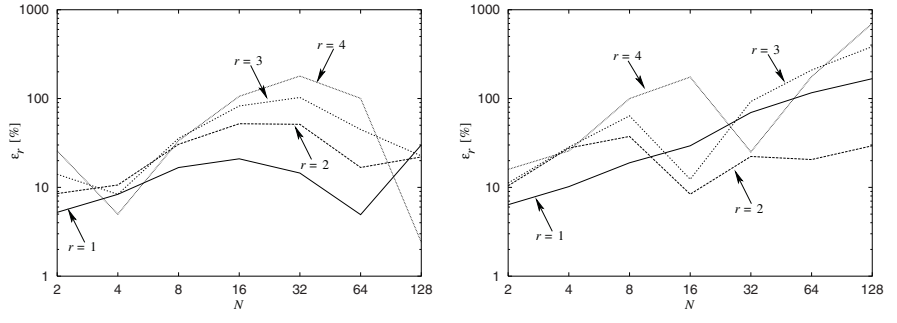

Fig. 5. ε_r [%] for N-ary or-parallel (.com: left and .nl: right)

In a binary or-parallel search we combine each possible combination of two domains and apply Eq. (15). The prediction error for the mean is in the percent range as given in Table 1 while for the higher moments the prediction error is moderate, since again, the workloads are no longer unimodal.

Table 1. ε_r [%] for empirical distributions

X_1	X_2	ε_1[%]	ε_2[%]	ε_3[%]	ε_4[%]
.cl	.com	4.0	8.5	21	73
.com	.nl	1.1	9.3	21	45
.nl	.cl	1.8	4.7	19	37

5 Conclusion

In this paper we have presented an analytical approach to predict the execution time distribution of N-ary and binary speculative parallel composition at a mere $O(1)$ solution complexity. Measurement results for uniform, exponential and normal distributions indicate an accuracy that lies in the percent range while for real workload distributions on internet search engines the prediction error is in the 10 percent range, provided workload unimodality.

References

1. Gautama, H. and van Gemund, A.J.C.: "Static performance prediction of data-dependent programs," in *ACM WOSP 2000*, Sept. 2000, pp. 216–226. Ottawa, Canada.
2. Gautama, H. and van Gemund, A.J.C.: "Low-cost performance prediction of data-dependent data parallel programs," in *Proc. of MASCOTS 2001*, IEEE Computer Society Press, Aug. 2001, pp. 173–182. Cincinnati, Ohio.
3. Gautama, H. and van Gemund, A.J.C.: "Performance prediction of data-dependent task parallel programs," in *Proc. of EuroPar 2001*, Aug. 2001, pp. 106–116. Manchester, United Kingdom.
4. Gautama, H.: "A statistical approach to performance prediction of speculative parallel programs," Tech. Rep. PDS-2003-007, Delft University of Technology, Delft, The Netherlands, May 2003.
5. Gelenbe, E., Montagne, E., Suros, R. and Woodside, C.M.: "Performance of block-structured parallel programs," in *Parallel Algorithms and Architectures* (Cosnard, M. *et al.*, (eds.)), Amsterdam: North-Holland, 1986, pp. 127–138.
6. Gumbel, E.J.: "Statistical theory of extreme values (main results)," in *Contributions to Order Statistics* (Sarhan, A.E. and Greenberg, B.G. eds.), New York: John Wiley & Sons, 1962, pp. 56–93.
7. Kruskal, C.P. and Weiss, A.: "Allocating independent subtasks on parallel processors," *IEEE TSE*, vol. 11, Oct. 1985, pp. 1001–1016.
8. Lester, B.P., "A system for the speedup of parallel programs," in *Proceedings of the 1986 Intl. Conference on Parallel Processing*, IEEE, Aug. 1986, pp. 145–152. Maharishi Intl. U, Fairfield, Iowa.
9. Liang, D.-R. and Tripathi, S.K.: "On performance prediction of parallel computations with precedent constraints," *IEEE TPDS*, vol. 11, no. 5, 2000, pp. 491–508.
10. Madala, S. and Sinclair, J.B.: "Performance of synchronous parallel algorithms with regular structures," *IEEE TPDS*, vol. 2, Jan. 1991, pp. 105–116.
11. Olsson, D.M. and Nelson, L.S.: "Nelder-Mead simplex procedure for function minimization," *Technometrics*, vol. 17, 1975, pp. 45–51.
12. Ramberg, J.S., Tadikamalla, P.R., Dudewicz, E.J. and Mykytka, F.M.: "A probability distribution and its uses in fitting data," *Technometrics*, vol. 21, May 1979, pp. 201–214.
13. Robinson, J.T.: "Some analysis techniques for asynchronous multiprocessor algorithms," *IEEE TSE*, vol. 5, Jan. 1979, pp. 24–31.
14. Sahner, R.A. and Trivedi, K.S.: "Performance and reliability analysis using directed acyclic graphs," *IEEE TSE*, vol. 13, Oct. 1987, pp. 1105–1114.
15. Schopf, J.M. and Berman, F.: "Performance prediction in production environments," in *Proceedings of IPPS/SPDP-98*, Los Alamitos, IEEE Computer Society, Mar. 30–Apr. 3 1998, pp. 647–653.
16. Sötz, F.: "A method for performance prediction of parallel programs," in *Proc. CONPAR 90-VAPP IV (LNCS 457)* (Burkhart, H., (ed.)), Springer-Verlag, 1990, pp. 98–107.
17. Stuart, A. and Ord, J.K.: *Kendall's Advanced Theory of Statistics*, vol. 1. New York: Halsted Press, 6 ed., 1994.
18. Trivedi, K.S.: *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. Englewood Cliffs, New Jersey: Prentice-Hall, 1982.