PERGAMON Computers and Mathematics with Applications 44 (2002) 1431–1437

An International Journal
computers & mathematics
with applications

www.elsevier.com/locate/camwa

Performance Analysis of a Parallel Dantzig-Wolfe Decomposition Algorithm for Linear Programming

JRJUNG LYU*
Department of Industrial Management Science
National Cheng Kung University
Tainan, Taiwan, R.O.C.
jlyu@mail.ncku.edu.tw

HSING LUH
National Cheng Chi University
Tainan, Taiwan, R.O.C.

MING-CHANG LEE
National Cheng Kung University
Tainan, Taiwan, R.O.C.

(Received October 2000; accepted August 2001)

Abstract—This paper employs the Dantzig-Wolfe decomposition principle to solve linear programming models in a parallel-computing environment. Adopting the queuing discipline, we showed that under very general conditions, the proposed algorithm speedup trends toward a limiting value as the number of processors increases.

Keywords—Linear programming, Decomposition algorithms, Parallel processing.

1. INTRODUCTION

Consider a linear programming (LP) problem that can be expressed in the following form:

$$\begin{aligned} \text{Minimize:} \quad & \mathbf{C}^{\mathsf{T}}\mathbf{X}, \\ \text{Subject to:} \quad & \mathbf{A}\mathbf{X} = \mathbf{b}, \\ & \mathbf{X} \geq \mathbf{0}. \end{aligned} \tag{1}$$

Suppose the $\mathbf{A}$ matrix has a special block-angular structure, namely,

$$\mathbf{A} = \begin{bmatrix} \mathbf{L}_1 & \mathbf{L}_2 & \dots & \mathbf{L}_n \\ \mathbf{A}_1 & & & \\ & \mathbf{A}_2 & & \\ & & \ddots & \\ & & & \mathbf{A}_n \end{bmatrix},$$

*Author to whom all correspondence should be addressed.

0898-1221/02/$ - see front matter Typeset by AMS-TeX
PII: S0898-1221(02)00267-5

where all $\mathbf{A}_i$ in the technology matrix $\mathbf{A}$ are independent blocks linked by coupling-equation matrices $\mathbf{L}_i$. In this research, a parallel algorithm, based on the Dantzig-Wolfe decomposition principle (DWDP), was developed to solve the linear programming problems as stated above. Since Dantzig and Wolfe developed the decomposition principle in the early sixties, this method is still widely adopted to cope with large-scale optimization problems. For instance, Ziarati, *et al.* [1] considered a multicommodity flow model for assigning locomotives to train-segments while employing DWDP to solve a very large-scale-scheduling problem. Desaulniers, *et al.* [2] used DWDP to solve the problem of choosing the best crew pairing at Air France. Other examples can be found in [3–5]. Given that larger and more complex mathematical models have become commonplace (see [6]), the importance of DWDP is well recognized by researchers.

An ideal Dantzig-Wolfe decomposition model considers a typical linear programming problem whose technological matrix is block-type. Since each block-type submatrix can be transformed into an independent subproblem while maintaining the global optimum, an ideal Dantzig-Wolfe decomposition algorithm represents a limiting case for the effects of parallelism. When a parallel Dantzig-Wolfe decomposition algorithm is developed, it can be shown that there is no wasted time associated with communication delays where the only source of communication delay is the actual time used in executing the program. This research, therefore, provides an upper bound for algorithm speedup using parallel processing for linear programs.

2. A PARALLEL LP ALGORITHM

In problem (1), let each $\mathbf{A}_i$ have m_i rows and p_i columns and each $\mathbf{L}_i$ be an $m_0 \times p_i$ matrix, for $i = 1, 2, \dots, n$. Letting $m = \sum_{i=0}^{n} m_i$ and $p = \sum_{i=1}^{n} p_i$, $\mathbf{A}$ is, therefore, an $m \times p$ matrix. By partitioning vectors $\mathbf{b}$, $\mathbf{X}$, and $\mathbf{C}$ into sizes corresponding to each $\mathbf{A}_i$, problem (1) can be rewritten as follows.

$$\begin{aligned}
\text{Minimize:} \quad & \sum_{i=1}^{n} \mathbf{C}_i^{\mathsf{T}} \mathbf{X}_i, \\
\text{Subject to:} \quad & \sum_{i=1}^{n} \mathbf{L}_i \mathbf{X}_i = \mathbf{b}_0, \\
& \mathbf{A}_i \mathbf{X}_i = \mathbf{b}_i, \\
& \mathbf{X}_i \geq \mathbf{0}.
\end{aligned} \tag{2}$$

We can then define the subproblem i, for i=1, 2, ... , n, as:

$$\begin{aligned}
\text{Minimize:} \quad & \left(\mathbf{C}_i^{\mathsf{T}} - \boldsymbol{\lambda}_0^{\mathsf{T}} \mathbf{L}_i\right) \mathbf{X}_i, \\
\text{Subject to:} \quad & \mathbf{A}_i \mathbf{X}_i = \mathbf{b}_i, \\
& \mathbf{X}_i \geq \mathbf{0},
\end{aligned} \tag{3}$$

where $\boldsymbol{\lambda}_0^{\mathsf{T}}$ is the vector denoting the simplex multipliers corresponding to the constraint $\sum_{i=1}^{n} \mathbf{L}_i \mathbf{X}_i = \mathbf{b}_0$.

In contrast to subproblem (3), problem (2) is called the master program. Based on the convexity properties of problems (2) and (3), which imply that all solutions can be written as a linear combination of their vertices, a two-level algorithm based on DWDP can be developed. In this algorithm, the master program is on the first level to search for the coefficients of the linear combination. Subproblem (3) is on the second level to solve the possible optimal vertices.

Consider that there exists a distributed computing environment (DCE), which has more than n independent workstations connected by a network and a centralized processor (or the master processor) to serve as the coordinator. Such a framework has proven to be a viable approach to provide concurrent computing power at reasonable costs [7]. Procedure 1 describes an algorithm based on the Dantzig-Wolfe decomposition principle (DWDP) that can be executed on the DCE.