

Closeout Notice Date 06-JUL-1998

Project Number E-24-X59

Doch Id 34600

Center Number 10/24-6-R8257-0A0

Project Director SAVELSBERGH, MATHIEU

Project Unit ISYE

Sponsor NATL SCIENCE FOUNDATION/GENERAL

Division Id 3393

Contract Number DMI-9410102

Contract Entity GTRC

Prime Contract Number

Title RIA: PARALLEL COMPUTING & MIXED INTEGER PROGRAMMING

Effective Completion Date 30-JUN-1998 (Performance) 28-SEP-1998 (Reports)

Closeout Action:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	N	
Final Report of Inventions and/or Subcontracts	N	
Government Property Inventory and Related Certificate	N	
Classified Material Certificate	N	
Release and Assignment	N	
Other	N	

Comments
LETTER OF CREDIT APPLIES. 98A SATISFIES PATENT REPORT.

Distribution Required:

Project Director/Principal Investigator	Y
Research Administrative Network	Y
Accounting	Y
Research Security Department	N
Reports Coordinator	Y
Research Property Team	Y
Supply Services Department/Procurement	Y
Georgia Tech Research Corporation	Y
Project File	Y

E-24-X59
#1 (New)

To NSF Program: DMS-Research Initiation Awards

APPENDIX VIII

Annual NSF Grant Progress Report

PI Name: Martin W. P. Savelsbergh

NSF Award Number: DMI-9410102

PI Institution: Georgia Institute of Technology

PI Address: 765 Ferst Drive, N. W.
Atlanta, GA 30332-0205

Date: 11/22/95

I certify that to the best of my knowledge (1) the statements herein (excluding scientific hypotheses and scientific opinions) are true and complete, and (2) the text and graphics in this report as well as any accompanying publications or other documents, unless otherwise indicated, are the original work of the signatories or individuals working under their supervision. I understand that the willful provision of false information or concealing a material fact in this report or any other communication submitted to NSF is a criminal offense (U.S. Code, Title 18, Section 1001.)

Signature: _____

Please include the following information:

1. A brief summary of overall progress, including results obtained to date, their relationship to the general goals of the award and their significance to science;
2. an indication of any current problems or favorable or unusual developments;
3. a brief summary of work to be performed during the next year of support if changed from the original proposal; and
4. any other information pertinent to the type of project supported by NSF or as specified by the terms and conditions of the grant.

If applicable, please attach a copy of any updated human subject or animal subject certification.
[Attach additional sheets as necessary.]

Parallel computing for mixed integer programming

Introduction

A great variety of practical problems in areas such as logistics and manufacturing can be formulated as mixed integer programs. Therefore, effective and efficient mixed integer programming software will have a significant impact on improving the planning and operation of various production and distribution processes. Unfortunately, the available software cannot solve many of the large-scale instances to (near) optimality in a reasonable amount of time. Parallel computing provides an opportunity for substantial progress in mixed integer programming and in this project I am exploring and exploiting these opportunities.

I am focusing on the development of an environment for the implementation of effective and efficient parallel linear programming based branch-and-bound algorithms for mixed integer programs. Topics that are being studied include reformulation, decomposition, row and column generation, row and column management, branching, parallel architectures, fine and coarse grained parallelism, task allocation, communication, and synchronization. The project also involves the development of a modular experimental code to test and evaluate the ideas and methodology that result from the basic research.

This project will result in a better understanding of the potential of parallel computing for mathematical programming and the implementation of special purpose algorithms for specific problems in logistics and manufacturing.

Mixed Integer Programming

As mixed integer programming problems are very difficult, optimization algorithms have to rely on some form of enumeration, usually branch-and-bound. In the past decade, research on linear programming based branch-and-bound methods for mixed integer programming problems has focused on improving the linear programming approximation. Reformulation techniques have been developed that have proven to be quite successful. As a result, two important variants of branch-and-bound have emerged: branch-and-cut and branch-and-price.

The basic idea of branch-and-cut is simple. Classes of valid inequalities are left out of the linear programming relaxation because there are too many constraints to handle efficiently and most of them will not be binding in an optimal solution anyway. Then, if an optimal solution to a linear programming relaxation is infeasible, a subproblem, called the separation problem, is solved to try to identify the violated inequalities in a class. If one or more violated inequalities are found, they are added to the linear program to cut off the infeasible solution. Then the linear program is reoptimized. Branching occurs when no violated inequalities are found to cut off an infeasible solution. Branch-and-cut allows separation and cutting to be applied throughout the branch-and-bound tree.

The philosophy of branch-and-price is similar to that of branch-and-cut except that the procedure focuses on column generation rather than row generation.

Parallel Computing

Parallel computing systems offer the promise of a quantum leap in the computing power that can be brought to bear on many important problems, including mixed integer programming problems.

Parallel computing systems consist of several processors that are located within a small distance of each other and that are specifically designed to jointly execute a computational task. There are several issues related to parallel computation that do not arise in serial computation. A first issue is the task allocation, that is, the breakdown of the total workload in smaller tasks that have to be assigned to different processors, and the proper sequencing of the tasks when some of them are interdependent and cannot be executed simultaneously. A second issue is the communication of interim results between processors. In many parallel algorithms the time spent on interprocessor communication is a sizable fraction of the total time needed to solve a problem. A third issue is the synchronization of the computations of different processors. In any parallel algorithm, it is necessary to coordinate, to some extent, the activities of the different processes. In synchronous methods, processors wait at predetermined points for the completion of certain computations or for the arrival of data. In asynchronous methods, there is no requirement for waiting at predetermined points.

Parallel Mixed Integer Programming

The observation that any two nodes of the search tree, each dealing with a subset of the solution space, can be solved independently, provides a natural basis for parallelization of branch-and-bound algorithms. Branch-and-cut as well as branch-and-price algorithms provide additional possibilities for parallelization:

1. Both branch-and-cut and branch-and-price algorithms require the solution of an optimization problem to generate violated rows and profitable columns respectively. In many cases, these problems are difficult and decomposable and could be solved by one or more dedicated processors.
2. Both branch-and-cut and branch-and-price algorithms require advanced row and column management functions to control the size of the active linear program. Cut pools and column pools could be maintained by a dedicated processor.

In this project, we are exploring the possibilities that parallelization has to offer specifically for branch-and-cut and branch-and-price algorithms.

Research Plan

To accomplish the goals set for this project, the following four phase research plan research plan has been adopted.

1. Development of a plain parallel LP based branch-and-bound algorithm for mixed integer programming.
2. Development of parallel branch-and-cut and parallel branch-and-price algorithms for specific optimization problems.
3. Experimentation with the parallel algorithms developed in the first phases to develop and analyze task allocation procedures, communication protocols, row and column management schemes, and tree search strategies.
4. Examination of the various components of state-of-the-art mixed integer programming solvers to determine whether they offer additional opportunities for parallelization.

Project status

The first phase of the research plan has been completed. A plain parallel LP based branch-and-bound algorithm for mixed integer programs has been implemented in C++ and has been tested on two different platforms: a network of workstations and an IBM SP/2 parallel computer. Because it is essential to minimize idle time and search overhead in order to achieve a high speedup, we have developed a flexible load balancing algorithm.

The second phase of the research plan is in progress. A parallel branch-and-price algorithm has been developed for the generalized assignment problem. With this algorithm, we have been able to solve instances with sizes that were impossible to solve with any of the existing sequential algorithms. We have also completed the design for a parallel branch-and-cut algorithm for mixed integer programs and are currently working on its implementation. The activities of this phase have already lead to important insights in the general characteristics of parallel branch-and-bound algorithms for mixed integer programming.

The third phase is in progress as well. We have designed four different row management schemes for branch-and-cut algorithms. The schemes differ in the storage location of the generated cuts: cuts are kept in local pools, in a global pool, in replicated global pools, or in a distributed global pool. Each of these row management schemes has different communication requirements. The choice of which schemes works best depends on the problem type, cut type, and architecture. We are in the process of implementing these schemes.

Parallel computing for mixed integer programming

Introduction

A great variety of practical problems in areas such as logistics and manufacturing can be formulated as mixed integer programs. Therefore, effective and efficient mixed integer programming software will have a significant impact on improving the planning and operation of various production and distribution processes. Unfortunately, the available software cannot solve many of the large-scale instances to (near) optimality in a reasonable amount of time. Parallel computing provides an opportunity for substantial progress in mixed integer programming and in this project I am exploring and exploiting these opportunities.

I am focusing on the development of an environment for the implementation of effective and efficient parallel linear programming based branch-and-bound algorithms for mixed integer programs. Topics that are being studied include reformulation, decomposition, row and column generation, row and column management, branching, parallel architectures, fine and coarse grained parallelism, task allocation, communication, and synchronization. The project also involves the development of a modular experimental code to test and evaluate the ideas and methodology that result from the basic research.

This project will result in a better understanding of the potential of parallel computing for mathematical programming and the implementation of special purpose algorithms for specific problems in logistics and manufacturing.

Mixed Integer Programming

As mixed integer programming problems are very difficult, optimization algorithms have to rely on some form of enumeration, usually branch-and-bound. In the past decade, research on linear programming based branch-and-bound methods for mixed integer programming problems has focused on improving the linear programming approximation. Reformulation techniques have been developed that have proven to be quite successful. As a result, two important variants of branch-and-bound have emerged: branch-and-cut and branch-and-price.

The basic idea of branch-and-cut is simple. Classes of valid inequalities are left out of the linear programming relaxation because there are too many constraints to handle efficiently and most of them will not be binding in an optimal solution anyway. Then, if an optimal solution to a linear programming relaxation is infeasible, a subproblem, called the separation problem, is solved to try to identify the violated inequalities in a class. If one or more violated inequalities are found, they are added to the linear program to cut off the infeasible solution. Then the linear program is reoptimized. Branching occurs when no violated inequalities are found to cut off an infeasible solution. Branch-and-cut allows separation and cutting to be applied throughout the branch-and-bound tree.

The philosophy of branch-and-price is similar to that of branch-and-cut except that the procedure focuses on column generation rather than row generation.

Parallel Computing

Parallel computing systems offer the promise of a quantum leap in the computing power that can be brought to bear on many important problems, including mixed integer programming problems.

Parallel computing systems consist of several processors that are located within a small distance of each other and that are specifically designed to jointly execute a computational task. There are several issues related to parallel computation that do not arise in serial computation. A first issue is the task allocation, that is, the breakdown of the total workload in smaller tasks that have to be assigned to different processors, and the proper sequencing of the tasks when some of them are interdependent and cannot be executed simultaneously. A second issue is the communication of interim results between processors. In many parallel algorithms the time spent on interprocessor communication is a sizable fraction of the total time needed to solve a problem. A third issue is the synchronization of the computations of different processors. In any parallel algorithm, it is necessary to coordinate, to some extent, the activities of the different processes. In synchronous methods, processors wait at predetermined points for the completion of certain computations or for the arrival of data. In asynchronous methods, there is no requirement for waiting at predetermined points.

Parallel Mixed Integer Programming

The observation that any two nodes of the search tree, each dealing with a subset of the solution space, can be solved independently, provides a natural basis for parallelization of branch-and-bound algorithms. Branch-and-cut as well as branch-and-price algorithms provide additional possibilities for parallelization:

1. Both branch-and-cut and branch-and-price algorithms require advanced row and column management functions to control the size of the active linear program. Cut pools and column pools could be maintained by a dedicated processor.
2. Both branch-and-cut and branch-and-price algorithms require the solution of an optimization problem to generate violated rows and profitable columns respectively. In many cases, these problems are difficult and decomposable and could be solved by one or more dedicated processors.

In this project, we are exploring the possibilities that parallelization has to offer specifically for branch-and-cut and branch-and-price algorithms.

Research Plan

To accomplish the goals set for this project, the following four phase research plan research plan has been adopted.

1. Development of a plain parallel LP based branch-and-bound algorithm for mixed integer programming.
2. Development of parallel branch-and-cut and parallel branch-and-price algorithms for specific optimization problems.
3. Experimentation with the parallel algorithms developed in the first phases to develop and analyze task allocation procedures, communication protocols, row and column management schemes, and tree search strategies.
4. Examination of the various components of state-of-the-art mixed integer programming solvers to determine whether they offer additional opportunities for parallelization.

Project status

The first phase of the research plan has been completed. A plain parallel LP based branch-and-bound algorithm for mixed integer programs has been implemented in C++ and has been tested on two different platforms: a network of workstations and an IBM SP/2 parallel computer. Because it is essential to minimize idle time and search overhead in order to achieve a high speedup, we have developed a flexible load balancing algorithm.

The second and third phase of the research plan are in progress. Basic parallel branch-and-cut and branch-and-price algorithms have been developed. The implementations rely heavily on the weak synchronization and consistency requirements of branch-and-cut and branch-and-price algorithms, which provide the flexibility that is necessary to be able to exploit parallelism successfully. For example, the correctness of a branch-and-cut algorithm does not depend on whether or not cuts are generated nor does it depend on when cuts are generated. Experimentation with these basic algorithms has increased our understanding of the issues in developing parallel implementations of branch-and-cut and branch-and-price algorithms.

The fundamental issue in parallelizing a branch-and-cut algorithm is the management of the set of cuts. A good distributed cut management scheme should aim to achieve the following goals simultaneously:

- Minimize total cut generation time over all processors. A careful design may allow communication to beat cut generation, thus reducing overall cut generation time.
- Maximize "useful sharing" of cuts. Since sharing of cuts involves communication, we would like to share only relevant cuts.
- Minimize latency of access to cuts. If there exists a relevant cut, the processes should be able to readily find it.
- Minimize bandwidth. We do not want to flood the communication system with cuts.

It is easy to see that these goals are inter-related, and tradeoffs have to be exploited by varying the degrees to which each is satisfied.

One straightforward implementation for distributed cut management consists of a central process that holds all the cuts, and acts as a server to all the other processes. The advantage of such a centralized scheme is the sharing of information. This sharing results in minimal duplication of effort in cut generation, and potentially in some synergism, because a cut generated at one processor may be found to be effective for nodes of another processor as well – this second processor benefits from the cut, which it may never have been able to generate itself. The disadvantage of this approach is high contention and latency; the central process can become a bottleneck and communication costs can be high. Clearly, this scheme does not scale well with the number of processors.

Another approach to distributed cut management is to make the cut pools fully distributed and independent, where every process maintains its own cut pool locally, and no sharing takes place. The advantage of the independent distributed sets is that the latency of access to a cut is almost constant, and it is bounded by the maximum cut generation time. However, at least two undesirable phenomena occur. First, duplication of effort takes place in terms of cut generation and memory requirements for storing the cuts. Second, since the cuts that are generated are typically globally valid, irrespective of the process at which they are generated, the advantage of sharing the global information is lost in a distributed implementation. Hybrid cut management schemes may be necessary to have a proper balance between the two extremes discussed above.

We have implemented a flexible cut management scheme that allows experimentation with the various schemes outlined above.

Branch-and-price poses many challenges to the parallel implementer as well. Should there be a single central pool of generated columns? Is there an advantage to sharing generated column information between processors? If so, how? If information is not shared, should the task scheduling algorithm be adjusted to take into account that some processors may be able to solve some problems faster than others, having already generated most of the applicable columns? Because columns are typically not globally valid, it may be better to have a subtree as a unit of work instead of a node, which is typically the case in branch-and-cut algorithms.

With all the necessary tools in place, in the remainder of the project we will complete and fine tune the cut and column management schemes, we will investigate how the parallel computing environment can be exploited to determine good feasible solutions early in the search process, and we will explore ways to take advantage of the startup phase in which not all processors are evaluating nodes of the search tree.

Relevant Papers

A. Atamturk, G.L. Nemhauser, and M.W.P. Savelsbergh. A Combined Lagrangian, Linear Programming and Implication Heuristic for Large-Scale Set Partitioning Problems, *Journal of Heuristics* 1, 247-259, 1996.

C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P. Vance. Branch-and-Price: Column Generation for Solving Huge Integer Programs, *Operations Research*, to appear.

R. Bixby, S. Ceria, C. McZeal, and M.W.P. Savelsbergh. An Updated Mixed Integer Programming Library: MIPLIB 3.0, LEC-96-03, Georgia Institute of Technology, 1996.

G. DePuy, M.W.P. Savelsbergh, J.C. Ammons, and L.F. McGinnis. An Integer Programming Heuristic for Component Allocation in Printed Circuit Card Assembly Systems, LEC-95-10, Georgia Institute of Technology, 1995.

Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Lifted Cover Inequalities for 0-1 Integer Programs: Computation, COC-94-09, Georgia Institute of Technology, 1994.

Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Lifted Cover Inequalities for 0-1 Integer Programs: Complexity, *INFORMS Journal of Computing*, to appear.

Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Sequence Independent Lifting, LEC-95-08, Georgia Institute of Technology, 1995.

Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Lifted Flow Cover Inequalities for 0-1 Mixed Integer Programs, LEC-96-05, Georgia Institute of Technology, 1996.

J. Linderoth, K. Perumalla, M.W.P. Savelsbergh. PARINO, a System for Parallel Mixed Integer Optimization, in preparation.

M.W.P. Savelsbergh. A Branch-and-Price Algorithm for the Generalized Assignment Problem, *Operations Research*, to appear.

M.W.P. Savelsbergh. Preprocessing and Probing for Mixed Integer Programming Problems, *ORSA Journal on Computing* 6, 445-454, 1994.

M.W.P. Savelsbergh and M. Sol. DRIVE: Dynamic Routing of Independent VEHICLES, *Operation Research*, to appear.

M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. A Time-Indexed Formulation for Single-Machine Scheduling Problems: Branch-and-Cut, LEC-95-02, Georgia Institute of Technology, 1995.

M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. A Time-Indexed Formulation for Single-Machine Scheduling Problems: Column Generation, LEC-96-04, Georgia Institute of Technology, 1996.

NSF Grant Conditions (Article 17, GC-1, and Article 8, FDP-II) require submission of a Final Project Report (NSF Form 98A) to the NSF Program Officer no later than 90 days after the expiration of the award. Final Project Reports for expired awards must be received before new awards can be made (NSF Grants Policy Manual Section 340).

Below, or on a separate page attached to this form, provide a summary of the completed projects and technical information. Be sure to include your name and award number on each separate page. See below for more instructions.

PART II - SUMMARY OF COMPLETED PROJECT (for public use)

The summary (about 200 words) must be self-contained and intelligible to a scientifically or technically literate reader. Without restating the project title, it should begin with a topic sentence stating the project's major thesis. The summary should include, if pertinent to the project being described, the following items:

- The primary objectives and scope of the project
- The techniques or approaches used only to the degree necessary for comprehension
- The findings and implications stated as concisely and informatively as possible

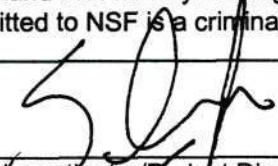
See attachment

PART III - TECHNICAL INFORMATION (for program management use)

List references to publications resulting from this award and briefly describe primary data, samples, physical collections, inventions, software, etc., created or gathered in the course of the research and, if appropriate, how they are being made available to the research community. Provide the NSF Invention Disclosure number for any invention.

see attachment

I certify to the best of my knowledge (1) the statements herein (excluding scientific hypotheses and scientific opinion) are true and complete, and (2) the text and graphics in this report as well as any accompanying publications or other documents, unless otherwise indicated, are the original work of the signatories or of individuals working under their supervision. I understand that willfully making a false statement or concealing a material fact in this report or any other communication submitted to NSF is a criminal offense (U.S. Code, Title 18, Section 1001).

	<i>11/24/97</i>
Principal Investigator/Project Director Signature	Date

**IMPORTANT:
MAILING INSTRUCTIONS**
Return this *entire* packet plus all attachments in the envelope attached to the back of this form. Please copy the information from Part 1, Block I to the *Attention block* on the envelope.

PART IV - FINAL PROJECT REPORT -- SUMMARY DATA ON PROJECT PERSONNEL

(To be submitted to cognizant Program Officer upon completion of project)

The data requested below are important for the development of a statistical profile on the personnel supported by Federal grants. The information on this part is solicited in response to public Law 99-383 and 42 USC 1885C. All information provided will be treated as confidential and will be safeguarded in accordance with the provisions of the Privacy Act of 1974. You should submit a single copy of this part with each final project report. However, submission of the requested information is not mandatory and is not a precondition of future award(s). Check the "Decline to Provide Information" box below if you do not wish to provide the information.

Please enter the numbers of individuals supported under this grant.

Do not enter information for individuals working less than 40 hours in any calendar year.

	Senior Staff		Post-Doctorals		Graduate Students		Under-Graduates		Other Participants ¹	
	Male	Fem.	Male	Fem.	Male	Fem.	Male	Fem.	Male	Fem.
A. Total, U.S. Citizens					/					
B. Total, Permanent Residents	/									
U.S. Citizens or Permanent Residents: ²										
American Indian or Alaskan Native....										
Asian.....										
Black, Not of Hispanic Origin.....										
Hispanic.....										
Pacific Islander.....										
White, Not of Hispanic Origin.....	/				/					
C. Total, Other Non-U.S. Citizens										
Specify Country										
1.										
2.										
3.										
D. Total, All participants (A + B + C)	/				/					
Disabled ³										

Decline to Provide Information: Check box if you do not wish to provide this information (you are still required to return this page along with parts I-III).

¹Category includes, for example, college and precollege teachers, conference and workshop participants.

²Use the category that best describes the ethnic/racial status to all U.S. Citizens and Non-citizens with Permanent Residency. (If more than one category applies, use the one category that most closely reflects the person's recognition in the community.)

³A person having a physical or mental impairment that substantially limits one or more major life activities; who has a record of such impairment; or who is regarded as having such impairment. (Disabled individuals also should be counted under the appropriate ethnic/racial group unless they are classified as "Other Non-U.S. Citizens.")

AMERICAN INDIAN OR ALASKAN NATIVE: A person having origins in any of the original peoples of North America and who maintains cultural identification through tribal affiliation or community recognition.

ASIAN: A person having origins in any of the original peoples of East Asia, Southeast Asia or the Indian subcontinent. This area includes, for example, China, India, Indonesia, Japan, Korea and Vietnam.

BLACK, NOT OF HISPANIC ORIGIN: A person having origins in any of the black racial groups of Africa.

HISPANIC: A person of Mexican, Puerto Rican, Cuban, Central or South American or other Spanish culture or origin, regardless of race.

PACIFIC ISLANDER: A person having origins in any of the original peoples of Hawaii, the U.S. Pacific territories of Guam, American Samoa, and the Northern Marinas; the U.S. Trust Territory of Palau; the islands of Micronesia and Melanesia; or the Philippines.

WHITE, NOT OF HISPANIC ORIGIN: A person having origins in any of the original peoples of Europe, North Africa, or the Middle East.

M.W.P. Savelsbergh
9410102

SUMMARY OF COMPLETED PROJECT

Parallel computing can significantly enhance our ability to solve large-scale mixed integer programs (MIPs). Since a great variety of practical problems in areas such as logistics and manufacturing can be formulated as mixed integer programs this will have a significant impact on our ability to effectively plan and operate various production and distribution processes.

Although parallelization of traditional linear programming (LP) based branch-and-bound algorithms is fairly straightforward, this is not the case with the more recent and more sophisticated branch-and-cut and branch-and-price algorithms. The reason being the fact that more information can and needs to be shared between the different processors.

Our research has demonstrated that the key to building effective and efficient parallel MIP solvers is to carefully analyze the importance of globally valid information generated during the solution process, such as conflict graphs, pseudo costs, and valid cuts, and, based on this analysis, to carefully design the algorithms that generate, use, and maintain this globally valid information.

Cut management in parallel branch-and-cut algorithms is an illustrative example. A flexible hybrid distributed cut management scheme, based on four types of cut pools, has been shown to outperform fully centralized and fully distributed cut management schemes on a variety of computing environments.

During the research project, we have developed PARINO (PARallel Integer Optimizer). PARINO is an easy-to-use environment for developing and testing parallel implementations of integer programming techniques.

M.W.P. Savelsbergh
9410102

TECHNICAL INFORMATION

Publications

1. M.W.P. Savelsbergh. A Branch-and-Price Algorithm for the Generalized Assignment Problem. *Operations Research*, to appear.
2. C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P.H. Vance. Branch-and-Price: Column Generation for Huge Integer Programs. *Operations Research*, to appear.
3. Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Cover Inequalities for 0-1 Linear Programs II: Complexity. *INFORMS Journal on Computing*, to appear.
4. A. Atamturk, G.L. Nemhauser, and M.W.P. Savelsbergh. A combined Lagrangian, linear programming, and implication heuristic for large-scale set partitioning problems. *Journal of Heuristics* 1, 247-259, 1996.
5. Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Sequence Independent Lifting of Cover Inequalities. E. Balas and J. Clausen (eds.). *Integer Programming and Combinatorial Optimization*. Lecture Notes in Computer Science 920, Springer-Verlag, 452-461, 1995.
6. J. Linderoth and M.W.P. Savelsbergh. A computational study of search strategies for mixed integer programming. Report LEC-97-12, Georgia Institute of Technology, 1997.
7. K. Perumalla, M.W.P. Savelsbergh, and U. Ramachandran. PARINO – An extendable framework for solving mixed integer programs in parallel. Report GIT-CG-97-07. Georgia Institute of Technology, 1997.
8. E.L. Johnson, G.L. Nemhauser, and M.W.P. Savelsbergh. Progress in Integer Programming: An Exposition. Report LEC-97-02, Georgia Institute of Technology, 1997.
9. Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Lifted Flow Cover Inequalities for Mixed 0-1 Integer Programs. Report LEC-96-05, Georgia Institute of Technology, 1996.
10. Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Sequence Independent Lifting. Report COC-95-08, Georgia Institute of Technology, 1995.
11. Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Cover Inequalities for 0-1 Linear Programs I: Computation. Report COC-94-09, Georgia Institute of Technology, 1994.