

# The Uncapacitated Time-Space Fixed-Charge Network Flow Problem: An Empirical Investigation of Procedures for Arc Capacity Assignment

Jeffery L. Kennington, Charles D. Nicholson

Department of Engineering Management, Information, and Systems, Southern Methodist University, PO  
Box 750123, Dallas, TX 75275-0123, USA {jlk@lyle.smu.edu, cd.nicholson@sbcglobal.net}

The nodes and arcs of a network configuration replicated over time is a common structure found in many applications, particularly in the area of logistics. A common cost structure for flows in arcs for such problems involves both a fixed and variable cost. Combining the two concepts results in the uncapacitated time-space fixed-charge network flow problem. These problems can be modeled as mixed binary linear programs and can be solved with commercial software. To create these models for uncapacitated arcs requires determining artificial arc capacities that are sufficiently large so that the solution space has not been altered, but are small enough that the linear programming relaxations are tight. In this investigation, we present a strategy for determining these artificial arc capacities for any time-space fixed-charge network flow problem. In extensive empirical tests, we provide statistical evidence that the strategy is superior to the usual techniques applied to this class of problem. Many of the most difficult problems were solved in only 5% of the computational time required by standard techniques.

*Key words:* network optimization, integer programming, logistics

*History:* Accepted by John N. Hooker, Area Editor for Constraint Programming & Optimization; received July 15, 2008; revised February 13, 2009; accepted May 22, 2009.

---

## 1. Introduction

A time-space network problem is one in which each node in a directed graph is associated with a specific time period. All arcs from time period  $t_1$  terminate in period  $t_2$ , where  $t_2 \geq t_1$ . That is, the flow is between nodes in the same time period or to nodes in a future time-period. Time-space networks are used to solve a variety of scheduling and logistics problems. The fixed-charge network flow problem is characterized by the inclusion of fixed and variable costs

on arcs and has a multitude of practical applications. Fixed-charge network flow problems are commonly used to model warehouse/plant location problems, transportation problems, network design problems, and others in which set-up costs exist. Combining the time-space and fixed-charge network paradigm defines the problem of interest for this investigation.

This study was motivated by an application of cash management faced by many large national banks. These banks manage cash distribution centers called vaults. A vault is simply a large warehouse where cash is counted, sorted, and packaged for redistribution. Deposits and withdrawals are made using armored vehicles which travel between vaults or to and from the nearest Federal Reserve Bank. The cost of transporting cash is modeled using a fixed and variable cost. Storing the cash in a vault overnight incurs only a variable cost. In the time-space fixed-charge (TSFC) network representation, illustrated in Figure 1, each business day is a distinct time period and the vaults are modeled as nodes replicated across the time periods (Frost et al. 2008). Deficits of cash at a given vault on a given day can be fulfilled by shipments from the nearest Federal Reserve Bank. Similarly, surplus cash can be removed from a bank's books by a deposit at the nearest Federal Reserve Bank. For the arcs illustrated in Figure 1, the fixed costs are relatively high compared to the variable costs.

Figure 1 About Here

## 2. The Literature

The only difference between the formulation of the fixed-charge problem and easy linear programming problems is the discontinuous cost function. This deceptively simple detail is sufficient to distinguish the fixed-charge problem as NP-hard (Garey and Johnson 1979). Solving the problem to optimality therefore becomes exponentially more difficult as the problem size increases. The fixed-charge problem has been studied extensively and is of significant practical use. It includes the single and multiple source fixed-charge transportation problem, capacitated and uncapacitated lot-sizing, warehouse location problem, and Steiner tree problem as variants. The literature is replete with different approaches to either improve algorithmic efficiency for optimal solutions or to develop heuristics that find good solutions quickly.

This literature review is divided into multiple sections. In the first, the history and development of exact approaches is examined. In the second, a review of various heuris-

tic approaches is presented. Finally, practical applications of the time-space problem are reviewed.

## 2.1. Exact Methods

Hirsch and Dantzig (1968) formulated the fixed-charge problem in 1954 and showed that the optimal solution will lie at an extreme point. Based on this finding, Murty (1968) devised the first exact procedure to solve the fixed-charge problem by ranking the extreme points. He found that it is not necessary to rank all the extreme points since it is possible to determine upper and lower bounds for the problem.

Gray (1971) devised a method to find the exact solution to the fixed-charge transportation problem. Gray's procedure, a branch-and-bound method which iteratively updates the problems bounds, was shown to work more efficiently than Murty's method for problems where the fixed-charges dominate. Steinberg (1970) introduced another branch-and-bound procedure that solves the fixed-charge problem exactly.

Kennington and Unger (1976) developed a more efficient method for solving the fixed-charge transportation problem. They formulated the problem as a network flow problem, used a linear relaxation as the lower-bound, and applied the "up" and "down" Driebeek-Tomlin penalties (Driebeek 1966, Tomlin 1971) to guide the branch-and-bound process. Computationally this method was shown to be substantially faster than Gray's method.

Barr et al. (1981) also studied the fixed-charge transportation problem. Their focus was to develop an exact approach that would work well on a sparse network since they claimed most real-world transportation problems are sparse. Their work enhanced the standard fathoming-tests and implemented an algorithm that calculated the Driebeek-Tomlin penalties more efficiently. Empirical tests demonstrated that the new algorithm solved both sparse and dense problems more efficiently than previous exact methods.

Cabot and Erenguc (1986) extended the work of Kennington and Unger (1976) and Barr et al. (1981) to create a second penalty that was used to select the branching variable in a general linear fixed-charge problem. They performed an empirical study comparing their new penalty with the Driebeek-Tomlin penalty used in the previous investigations. In most of their cases, they found that the new penalty was stronger than the Driebeek-Tomlin penalty.

Palekar et al. (1990) developed another penalty to solve the fixed-charge transportation problem. They claim that this conditional penalty is stronger than both the Driebeek-Tomlin

and the Cabot-Erenguc penalties. Their computational work showed that for complex problems the penalties reduced the number of transportation subproblems by up to 65% over the Cabot-Erenguc method. Furthermore, they provided guidelines to classify the difficulty level for specific instances of problems. Citing Kennington (1976), they determined that the F/C ratio, the ratio of fixed-charges to continuous costs, is of critical importance to determine the difficulty of a problem. However, since the F/C ratio is not available until after the problem is solved, they defined an a priori approximation of the ratio which allows difficulty to be estimated before solving the problem. Lamar and Wallace (1997) revised the Palekar conditional penalties noting that the underlying philosophy was correct, but that a simple revision was necessary to ensure an optimal solution is found. The revised penalties are empirically shown to still dominate the Driebeek-Tomlin penalties. Bell et al. (1999) extended this work deriving another conditional penalty and new capacity improvement techniques for the problem by using concave relaxation during the branch-and-bound procedure instead of the standard linear relaxation. Their computational work shows a significant increase in efficiency for solving hard problems.

Herer et al. (1996) studied the single-sink fixed-charge transportation problem and developed new rules that increase the efficiency of the implicit complete enumeration algorithm proposed by Haberl (1991). They introduced two domination rules and two lower bound rules which they determined can be combined without interfering with each other. Empirically they demonstrated that for all challenging problems, the combined rule algorithm significantly outperformed all other algorithms tested. However, they noted that for small problems, such as may be encountered when the single-sink fixed-charge transportation problem is a sub-problem, the preprocessing time necessary to implement the rule combination algorithm may outweigh the performance improvements.

Alidaee and Kochenberger (2005) also studied the single-sink fixed-charge transportation problem and presented a dynamic programming approach. The theoretical complexity of their algorithm is significantly better than the enumeration techniques of Herer et al. (1996).

Cruz et al. (1998) used branch-and-bound methods to solve the uncapacitated fixed-charge network flow problem. Instead of the standard linear relaxation to provide lower bounds to the problem, they used Lagrangean relaxation in its place. This relaxation in itself does not create lower bounds that are tighter than the linear relaxation, but the formulation allowed them to apply a heuristic that does produce better lower bounds and efficient upper bounds throughout the procedure. Their computational experiments validated

their method making faster solutions possible on larger problems. Kim and Hooker (2002) investigated a method that combines constraint programming with optimization techniques to solve the fixed-charge problem. They found that for certain fixed-charge variants, their hybrid approach was superior to cutting-plane techniques.

Ortega and Wolsey (2003) used a branch-and-cut algorithm to solve the uncapacitated fixed-charge problem to optimality. Their procedure utilizes several heuristics including a greedy algorithm to generate good node subsets for the dicuts and the dynamic slope-scaling heuristic developed by Kim and Pardalos (1999). They claim that their method used in combination with the cutting-plane algorithms already incorporated in commercial solvers increases solution efficiency. Computational experiments on several types of fixed-charge problem instances validated their claims.

Costa (2005) presents a literature review of fixed-charge network design problems that appear to be well suited for application of the Benders decomposition algorithm. That is, each of these problems decompose into an integer program with one continuous variable and a network auxiliary problem. Since the auxiliary problems are easy, he claims that this approach is one of the most appropriate procedures for this class of problems. Due to the success of the investigations reviewed, he believes that this approach will gain additional acceptance in the future.

## **2.2. Heuristic Methods**

Cooper and Drebes (1967), Denzler (1969) and Steinberg (1970) each proposed heuristics for the fixed-charge problem based on an adjacent extreme point search. Walker (1976) independently developed a similar search technique. The algorithm, known as SWIFT, has two phases. The first is identical to the LP simplex procedure except that the rule for selecting the nonbasic variable to enter the basis is modified. The second phase forces nonbasic vectors into the basis to move the algorithm to a non-optimal adjacent extreme point. Phase 1 is then repeated from the new extreme point in an attempt to move the algorithm away from the first local optimum to find an improved one. The test cases, while small problems by current standards, showed the superiority of SWIFT over the previous approximate methods.

Billheimer and Gray (1973) examined a special case of the fixed-charge problem, the single-source, single-sink multicommodity transshipment problem. Their procedure alternatively adds and drops arcs from the network problem based on certain criteria until a local

optimum is reached.

Balakrishnan et al. (1989) studied the uncapacitated network design problem, another variation of the fixed-charge problem, by using the dual-ascent technique which solves the dual of the problem approximately. After obtaining a solution, an add-drop heuristic was implemented. The joint algorithm was shown to quickly solve large problems, producing solutions within 4% of the optimum in most cases. The authors note however, that the dual-ascent procedure is not effective in the capacitated version of the network design problem.

Tabu search is a metaheuristic for solving optimization problems designed to guide other search procedures to prevent them from becoming trapped at a local optimum (Glover 1990). This is accomplished by a rule inherent to the method which forbids certain search movements for a set number of iterations known as the tabu tenure. Sun and McKeown (1993) applied tabu search to the fixed-charge problem. The tabu conditions used impose a tabu tenure on nonbasic variables entering the basis, and a separate rule that ensures variables brought into the basis remain basic for a certain number of iterations. Additionally, they implemented a criterion which allows the procedure to override the tabu conditions, and a function that diversifies the search by forcing variables into the basis that have been nonbasic for the longest time. In all of their test cases, the tabu search either found the optimal solution or solutions very close to optimal. Tabu search has also been effective solving the fixed-charge transportation problem (Sun et al. 1998). In both investigations, the authors note that further research is necessary to determine the best tabu parameters to solve difficult problems.

Kim and Pardalos (1999) introduced the dynamic slope scaling procedure to approximate the fixed-charge problem. This procedure attempts to replace the non-linear objective function of the fixed-charge problem with a linear factor that simultaneously reflects the variable and fixed costs of the problem. The linear factor is found by solving a series of LP problems and recursively updating the cost function in each iteration. In their test cases, medium-sized problems were solved within 0.65% of optimal. For larger problems, ranging in size from 440 to 10,200 arcs, solutions were achieved quickly, with the longest running solve time being 72 seconds.

The more-for-less phenomena are paradoxical situations in which more quantity can be shipped through a transportation network for the same or lower costs. Adlakha et al. (2007) developed a heuristic that identifies the more-for-less phenomenon in fixed-charge transportation problems. The heuristic requires a basic feasible solution as an initial point and

their experimental results demonstrate its success of finding more-for-less solutions when initialized at either optimal or non-optimal solutions.

Recently a significant quantity of research has focused on genetic algorithms to tackle the fixed-charge problem. Genetic algorithms, introduced by Holland (1975) are probabilistic, adaptive algorithms that attempt to evolve a population of candidate solutions known as chromosomes, towards a global optimal. The evolution takes place through a series of genetic operations such as mutation and crossover which produce a different population pool in the subsequent generation. Mutation is a genetic operator that randomly selects and modifies solutions within the population. Crossover is an operation which combines two different solutions to produce two similar offspring.

Michalewicz et al. (1991) applied genetic algorithms to the non-linear transportation problem and developed techniques for initialization, crossover, and mutation that maintain population feasibility. Their work showed that for problems with discontinuous cost functions, genetic algorithms outperform the gradient-constrained non-linear solver implemented in MINOS. Jaramillo et al. (2002) used genetic algorithms to solve the location problem, including capacitated and uncapacitated fixed-charge problems. The authors conclude that genetic algorithms are promising for the uncapacitated fixed-charge problem since their test cases showed it quickly converged to a good solution. However, for the capacitated fixed-charge problem they found the genetic algorithm to perform poorly and subsequently do not recommend it as a viable method.

Gen and Cheng (2003) provide a survey of genetic algorithms applied to difficult network design problems. They claim that a genetic algorithm approach is very effective for many kinds of problems including the fixed-charge transportation problem. They conclude that a key issue for solving problems successfully is defining an encoding that matches the nature of a given problem.

Sheng et al. (2006) used a spanning-tree encoded genetic algorithm to solve transportation problems with a discontinuous piecewise linear cost function. The fixed-charge transportation problem is a simplified variant of their problem, since its cost function has only one linear segment. They conclude that their method attains better quality solutions more efficiently than the matrix-based genetic algorithm used in Michalewicz et al. (1991).

Additional research applying genetic algorithms to fixed-charge problems, may be found in Gottlieb and Paulmann (1998), Gen et al. (2001), Duhamel (2001), Gen et al. (2005), Tohyama et al. (2006), Sheng and Dachen (2006), and Jo et al. (2007). Yang and Feng

(2007) studied the stochastic version of the fixed-charge transportation problem, in which the supplies, demands, arc-capacities, linear and fixed costs may be represented by random variables. They formulated the problem as three different goal-programming models: (i) expected-value goal programming, (ii) chance-constrained goal programming, and (iii) dependent-chance goal programming. The goal programming models were solved by a hybrid technique which combines tabu search with simulation. Numerical examples demonstrated their procedure was robust relative to different algorithm parameters, such as tabu tenure and stopping criteria.

Nahapetyan and Pardalos (2008) reformulated the fixed-charge network flow problem as a concave piecewise linear network flow problem, replacing the cost function with two linear pieces. They show that for an appropriate selection of the linear pieces, the reformulated problem is equivalent to the original problem. The correct selection of the linear pieces is difficult, and the authors solved a series of bilinear relaxations, modifying the linear cost function each time, until a stopping criteria is reached and an approximate solution is found. Empirically they found this method to outperform the dynamic-slope scaling procedure of Kim and Pardalos (1999) in both solution quality and CPU time.

### **2.3. Time-Space Problems**

Time-space network models are used in a variety of practical transportation problems. Zawack and Thompson (1987) used a time-space model to represent city traffic flow through a capacitated road system having both one-way and two-way streets. Powell (1986) and Frantzeskakis and Powell (1990) made use of time-space models to solve the stochastic vehicle allocation problem. Gintner et al. (2005) and Kliewer et al. (2006) applied the model to bus scheduling problems. Haghani and Oh (1996) and Yan and Shih (2007) used time-space networks to produce solutions to logistic problems encountered in disaster relief management. Train routing and scheduling is an important application area which frequently employs time-space network modeling (Florian et al. 1976, Sherali and Tuncbilek 1997, Ahuja et al. 2005). Wu et al. (2005) used time-space modeling to solve a fleet-sizing problem motivated by the truck-rental industry. Chardaire et al. (2005) introduced and examined the convoy movement problem, a problem motivated by military applications in which it is especially important to solve quickly and efficiently. The authors accomplished this by using Lagrangean relaxation on a time-space formulation of the problem. Time-space models have also been effectively utilized in the airline industry for flight scheduling (Jarrah et al. 1993), flight scheduling due

to airport closures (Yan and Lin 1997, Thengvall et al. 2003), crew scheduling (Guo et al. 2006) and airplane taxi planning (Marín 2006, Marín and Codina 2008).

### 3. Constructing a Tight LP Relaxation

Fixed-charged problems require binary and continuous variables to model variable and fixed costs associated with flow on an arc. A constraint is used for each arc that forces the binary variable to assume a value of 1 whenever the corresponding continuous variable assumes a value greater than zero. This constraint requires parameters whose values are sufficiently large so that the feasible region is not reduced. The objective of this investigation is to determine small, yet adequate values which ensure the optimal solution can be attained. The procedure leverages the time-space structure of the TSFC problem to generate logical choices for the parameters. Extensive computational testing will quantify the performance improvements gained by using this procedure when solving TSFC problems exactly.

Let  $N = \{1, \dots, n\}$  be the set of spatial nodes,  $T = \{0, \dots, t\}$  be the set of time periods and  $\bar{N}$  denote the set of node-time combinations in the time-space network. That is, the elements of  $\bar{N}$  are  $(i, r)$  pairs where  $i \in N, r \in T$ . Let  $A$  denote the set of arcs in the network. Each element of  $A$  is a 4-tuple,  $(i, r, j, s)$  where  $(i, r) \in \bar{N}, (j, s) \in \bar{N}$ , and  $r \leq s$ . Let the decision variable  $x_{irjs}$  denote the flow on arc  $(i, r, j, s) \in A$  and the parameter  $M_{irjs}$  be a number whose value is at least the optimal flow value on  $(i, r, j, s)$ . The cost per unit flow on arc  $(i, r, j, s)$  is  $c_{irjs}$ , the fixed cost on arc  $(i, r, j, s)$  is  $f_{irjs}$ , and  $y_{irjs}$  is a binary variable which will take on a value of 1 if  $x_{irjs} > 0$ ; and 0, otherwise. We assume that both  $c_{irjs}$  and  $f_{irjs}$  are nonnegative for all  $(i, r, j, s) \in A$ . The parameters  $R_{ir}$  are the requirements at node  $(i, r) \in \bar{N}$ . If  $R_{ir} > 0$  then node  $(i, r)$  is a supply node, if  $R_{ir} < 0$  then node  $(i, r)$  is a demand node, otherwise node  $(i, r)$  is a transshipment node. Given a directed graph  $G = (\bar{N}, A)$  the time-space fixed-charge network flow model is formally stated as follows.

$$\min \sum_{(i,r,j,s) \in A} (c_{irjs}x_{irjs} + f_{irjs}y_{irjs}) \quad (1)$$

$$\text{s.t.} \quad \sum_{(i,r,j,s) \in A} x_{irjs} - \sum_{(j,s,i,r) \in A} x_{jsir} = R_{ir} \quad \forall (i, r) \in \bar{N} \quad (2)$$

$$0 \leq x_{irjs} \leq M_{irjs}y_{irjs} \quad \forall (i, r, j, s) \in A \quad (3)$$

$$y_{irjs} \in \{0, 1\} \quad \forall (i, r, j, s) \in A \quad (4)$$

The constraints (3) ensure that the fixed cost is incurred if  $x_{irjs} > 0$ . Clearly,  $M_{irjs}$  must be large enough so that it does not create an artificial constraint on the flow. Choosing a value for  $M_{irjs}$  that is too large will not provide a tight linear relaxation for use in a branch-and-bound algorithm. Hence, our objective is to choose the smallest possible values for  $M_{irjs}$  such that the solution space remains unchanged.

Let P1 denote the problem (1)-(4) with,

$$M_{irjs} = \infty \quad \forall (i, r, j, s) \in A$$

Since the arc flows in a TSFC do not flow to earlier time periods, the total demand in an arbitrary time period  $s$  must be met by the total supply available at time  $s$ . Let  $\hat{T} = \{0, \dots, s-1, \}$ . Let  $(j, s)$  be a node in  $\bar{N}$  and  $\Gamma_{js}$  denote the absolute maximum amount of flow that could be sent through any arc with head node  $(j, s)$ . Then,

$$\Gamma_{js} = \sum_{i \in N, r \in \hat{T}} R_{ir} + \sum_{i \in N/j: R_{is} > 0} R_{is} \quad \forall (j, s) \in \bar{N} \quad (5)$$

Let P2 denote the problem (1)-(4) with,

$$M_{irjs} = \Gamma_{js} \quad \forall (i, r, j, s) \in A$$

The calculation of  $M_{1222}$  and  $M_{3222}$  is illustrated in Figure 2. The excess supply in period  $t = 0$  is 15, the excess supply in period  $t = 1$  is 2 and  $R_{12}$  is 4 for a total of 21 units. Hence, the maximum flow in either arc  $(1, 2, 2, 2)$  or  $(3, 2, 2, 2)$  is 21.

Figure 2 About Here

**Proposition:** Suppose P1 has a feasible solution,  $c_{irjs} \geq 0$  and  $f_{irjs} \geq 0$  for all  $(i, r, j, s) \in A$ , and  $x^*$  is an optimum for P1, then  $x^*$  is an optimum for P2.

**Proof:** Since the feasible region for P2 is a subset of the feasible region for P1, we only need to show that  $x^*$  is feasible for P2. Let  $x_{ir\hat{j}s}^*$ , denote the flow on arc  $(\hat{i}, r, \hat{j}, s)$ . Due to (2) the flow into period 1 from period 0 is equal to:

$$\sum_{i \in N} R_{i0}$$

Likewise due to (2) the flow into period 2 from period 1 is equal to:

$$\sum_{i \in N} (R_{i0} + R_{i1})$$

Hence, the flow into period  $s > 0$  from period  $s - 1$  is equal to:

$$\Omega_s = \sum_{i \in N} (R_{i0} + R_{i1} + \cdots + R_{is-1}) \quad (6)$$

Consider only the requirements at time  $s$  and let  $z_{ij}^s$  denote the flow from node  $i$  to node  $j$  of those requirements at time  $s$ . That is,  $z_{ij}^s = x_{ijs}$  when  $\Omega_s = 0$ . Let  $\Delta_j$  denote the sum of supply at  $s$  and is defined as,

$$\Delta_j = \sum_{k \in N: R_{ks} > 0} R_{ks}$$

Since  $c_{irjs} \geq 0$  and  $f_{irjs} \geq 0$ , then

$$z_{ij}^s \leq \begin{cases} \Delta_j - R_{js}, & \text{if } R_{js} > 0 \\ \Delta_j, & \text{otherwise} \end{cases} \quad (7)$$

Combining (6) with (7) implies:

$$x_{irjs}^* \leq \Omega_s + z_{ij}^s = \Gamma_{js}$$

Thus  $x_{irjs}^*$  is feasible for P2. Therefore,  $x^*$  is an optimum for P2.  $\square$

## 4. Empirical Evaluation

The purpose of the computational experiments described in this section is to compare branch-and-bound performance when using two different methods for generating the  $M_{irjs}$  values while solving TSFC problems exactly. The first method (M1) is a common method for determining the parameters. For each arc,  $M_{irjs}$  is set equal to the sum of the total supply within the network. This obviously ensures that the value is large enough to not induce artificial bounds. The second method (M2) will use the procedure described in Section 3.

Two classes of problems were attempted. The first class includes all possible arcs between nodes within a time period as well as all possible arcs to nodes in the next time period. The second class is again fully dense within a time period, but includes arcs from each node in a time period to all nodes in all future time periods.

All node requirements, variable costs, and fixed costs for each test problem were randomly generated from a uniform distribution with pre-specified lower and upper limits. If possible, the demand at the last demand node was set so that total supply equaled total demand. Due to the structure of the TSFC, it is easy to determine if a problem has a feasible solution and infeasible problems were discarded. That is, for any time period  $r \in T$ , if the total demand in time periods  $0, \dots, r$  exceeds the total supply in time periods  $0, \dots, r$  then (1)-(4) has no feasible solution. For  $r \in T$ , let

$$\alpha_r = \sum_{i \in N} R_{ir}$$

and

$$\beta_r = \sum_{s \in T: s \leq r} \alpha_s$$

If  $\beta_r < 0$  for any  $r \in T$ , then the problem is discarded.

Three range levels, corresponding to high, medium and low values, were generated for the node requirements. Similar levels were established for determining the variable and fixed costs. A problem with the high level of requirements, the medium level of variable costs, and the low level of fixed costs is denoted as an HML problem type. A full factorial design was used to test all possible combinations of high, medium, and low settings, resulting in 27 problem sets. For both problem classes and for both methods, thirty test problems were attempted for each of the 27 sets, resulting in a total of 3,240 test problems. All problems have 5 nodes replicated across 6 time-periods. For the first class of problems, each has 275 constraints, 245 binary variables, and 245 continuous variables. For the second class of problems, each has 525 constraints, 445 binary variables, and 445 continuous variables. The characteristics of the test problems are summarized in Table 1.

Table 1 About Here

The problems were solved using CPLEX 10.0 on Dell PE 2950 Dual Quad Core, 2.6GHz, 32GB RAM machines running Linux CentOS 4.6. The maximum time per instance was fixed at 4 CPU hours and the relative MIP gap tolerance was set to 0.0. The integrality parameter was left at the default value  $10^{-5}$ . Several problems did not solve within the specified time

limit. A few problems did not achieve an optimal solution due to the integrality settings in CPLEX. The majority of those early terminations occurred while using the M1 procedure. Only problems solved exactly by both methods were included in the averages and used to compute the comparison factors.

For both problem classes, there is a statistically significant difference in performance between the problem types. Figure 3 illustrates that problems with a lower ratio of variable to fixed cost often solve much faster than other problems. The fastest problem sets are those with the high level of variable cost and the low level of fixed cost. Problems with the the same level for variable and fixed costs have the most variability in the observed solve times. Tables 2 and 3 show the percentage of problems solved to optimality, average number of pivots, branch-and-bound nodes, and CPU time used to solve the test problems. Each of the problems in a given set has equal weight in determining the averages reported in Tables 2 and 3. The average pivots ratio, average branch-and-bound nodes ratio, and average CPU time ratio are also reported. The average pivots ratio is defined as the average of the pivots ratios in a problem class for the instances which were solved to optimality by both strategies. For example, there were 23 LLL problem instances solved to optimality by methods M1 and M2. The resulting 23 pivot ratios observed are averaged together to produce the statistic reported in Tables 2 and 3. A similar calculation was performed for the average branch-and-bound nodes ratio and average CPU time ratio.

---

Figure 3, Tables 2 and 3 About Here

For the class 1 problems, method M1 solved about 83% of the 810 problems attempted within the 4 hour time limit, while M2 solved 94%. Easy problems, those that require less than 1 second to complete, do not benefit significantly from the M2 procedure. However, the vast majority of the harder problems did see significant reductions in pivots, branch-and-bound nodes, and CPU time. For some hard problem instances the performance boost is dramatic. Figure 4 illustrates how the M1/M2 solve time ratio increases as the problems become more difficult. The log-log graph clearly shows a marked difference in the time ratio for problems when the M1 solve time is greater than 1 second. Statistically, at a 95% confidence level, there is sufficient evidence to conclude that M2 outperforms M1 for these harder problems.

---

Figure 4 About Here

Method M2 solved test problems up to 53 times faster than M1, using up to 45 times fewer pivots and 35 times fewer branch-and-bound nodes required by method M1. Of the hard problems successfully solved by both methods, M2 was the faster method in 351 of 370 instances. In problem types MLM, MLH, MMH and HLH, method M1 solved less than a third to optimality, ranging from 20% to 33%. Method M2 solved more than twice as many, ranging from 53% to 87% solved exactly.

The HLH problem results contain a notable performance discrepancy. The problem set contains a single test instance, problem P100, in which M1 drastically outperforms M2. Method M1 solved problem P100 twice as fast as method M2, using only 25% of the pivots, and 13% of the branch-and-bound nodes required by M2. Excluding this problem, M2 solved the 9 remaining HLH problems from 2.4 to 14 times faster than M1.

There are a small number of other hard test problems in which method M1 produces an exact solution more efficiently than method M2. In terms of the solve time ratio, the most notable of these is found in the problem type LLH, problem P1800. Method M1 solves it in less than 10 seconds, while method M2 takes more than 3 times as long to produce the same answer. Since the branch-and-bound algorithm uses a set of clever heuristics to search the feasible region, it is not surprising that method M1 occasionally outperforms method M2. That is, there is some luck involved in finding a good upper bound early in the search that eliminates searching over the unattractive branches in the tree.

Class 2 problems, having nearly twice the number of variables and twice the number of constraints, were considerably harder problem instances. Method M1 solved only 62% of all the problems attempted while method M2 solved 76%. The class 2 results support the observation that easy problems do not benefit much from the M2 procedure. Some of the hard class 2 problems saw sizeable performance improvements. Method M2 solved test problems up to 225 times faster than M1, used up to 151 times fewer pivots and 107 times fewer branch-and-bound nodes required by method M1. Method M2 produced optimal solutions faster than M1 in 290 of 327 hard problems. In Figure 5, the class 2 problem M1/M2 solve time ratios are plotted against the M1 solve times on a log-log scale. There is considerable variability in performance improvements as the problems become harder, but the trend is generally positive.

Similar to the class 1 results, problem types MLM, MLH, MMH and HLH were difficult to solve within the time limit. Method M1 solved less than 10%, while M2 solved 26% of these problem types. There are also a few noteworthy exceptions in the class 2 set of problems.

For example, problem P1600 of the LML set was solved by the M1 procedure in 23 minutes, which was less than half of the time required by M2.

Figure 5 About Here

Another computational experiment was conducted with supply and demand values randomly generated from a Normal distribution as opposed to a Uniform distribution. The most difficult problem types were selected for this evaluation (i.e. LLL, LLM, LLH, LMM, LMH, etc). The Normal distributions were created such that 90% of the distribution was contained between the original minimum and maximum values used for the Uniform distribution. These values are shown in Table 4. Using these distributions, thirty-six test cases were attempted and the results are recorded in Table 5. As expected, many of these problems were not solved in the four hour time limit. For those that were solved using both strategies, the M2 strategy always resulted in a smaller solution time. These results are consistent with those obtained using the Uniform distribution.

Tables 4 and 5 About Here

## 5. Summary and Conclusions

A strategy was presented to improve the continuous relaxation of the uncapacitated time-space fixed-charge network problem. Our method leverages the unique structure that a time-space problem imposes on a fixed-charge problem. Of critical importance, the method was proven to not create artificial bounds in a TSFC network. Extensive experimentation compared this procedure with a common alternative method used to calculate  $M_{i,r,j,s}$  parameters in fixed-charge problems. The computational work validated the hypothesis that the new method could significantly improve performance time for exact solutions. Specifically for hard problems, those that require more than 1 second of CPU time, the new method solved 90% to 95% of them faster than the competing method. For the most difficult problems solved, our new strategy resulted in a speed-up factor of approximately 20 to 1.

Easy problems did not benefit from the use of the procedure. Statistically, there was no difference in average solve times for these easier problems. Further it was noted that the variable to fixed costs relationship was an important factor in determining problem difficulty. Problems with higher level settings for fixed costs than variable costs resulted in substantially

longer average run times. The simplicity of the procedure combined with the success of the experimentation suggests that the proposed method can be very useful in solving difficult TSFC problems, and related problems such as those described by Frost et al. (2008).

## Acknowledgments

This research was supported in part by the Office of Naval Research under Award No. N00014-07-1-0192.

## References

- Adlakha, V., K. Kowalski, R. Vemuganti, B. Lev. 2007. More-for-less algorithm for fixed-charge transportation problems. *Omega* **37** 116–127.
- Ahuja, R. K., J. Liu, J. B. Orlin, D. Sharma, L. A. Shughart. 2005. Solving real-life locomotive-scheduling problems. *Transportation Science* **39** 503–517.
- Alidaee, B., G. A. Kochenberger. 2005. A note on a simple dynamic programming approach to the single-sink, fixed-charge transportation problem. *Transportation Science* **39** 140 – 143.
- Balakrishnan, A., T.L. Magnanti, R.T. Wong. 1989. Dual-ascent procedure for large-scale uncapacitated network design. *Operations Research* **37** 716 – 740.
- Barr, Richard S., Fred Glover, Darwin Klingman. 1981. A new optimization method for large scale fixed charge transportation problems. *Operations Research* **29** 448–463.
- Bell, G., B. Lamar, C. Wallace. 1999. Capacity improvement, penalties, and the fixed charge transportation problem. *Naval Research Logistics Quarterly* **46** 341–355.
- Billheimer, J. W., Paul Gray. 1973. Network design with fixed and variable cost elements. *Transportation Science* **7** 49 – 74.
- Cabot, A. Victor, S. Selcuk Erenguc. 1986. Improved penalties for fixed cost linear programs using Lagrangean relaxation. *Management Science* **32** 856 – 869.
- Chardaire, P., G. P. McKeown, S.A. Verity-Harrison, S.B. Richardson. 2005. Solving a time-space network formulation for the convoy movement problem. *Operations Research* **53** 219 – 230.
- Cooper, L., C. Drebes. 1967. An approximate solution method for the fixed charge problem. *Naval Research Logistics Quarterly* **8** 101–113.
- Costa, A. M. 2005. A survey on Benders decomposition applied to fixed-charge network design problems. *Computers and Operations Research* **32** 1429 – 1450.

- Cruz, F. R. B., J. MacGregor Smith, G. R. Mateus. 1998. Solving to optimality the uncapacitated fixed-charge network flow problem. *Computers and Operations Research* **25** 67–81.
- Denzler, D. 1969. An approximate algorithm for the fixed charge problem. *Naval Research Logistics Quarterly* **16** 411–416.
- Driebeek, N. 1966. An algorithm for the solution of mixed integer programming problems. *Management Science* **12** 576–587.
- Duhamel, Christophe. 2001. Solving the uncapacitated fixed-charge network flow with metaheuristics. *Proceedings of the 4th Metaheuristics International Conference, Porto, Portugal, July 2001*. 685–689.
- Florian, M., G. Bushell, J. Ferland, G. Guerin, L. Nastansky. 1976. Engine scheduling problem in a railway network. *Canadian Journal of Operational Research and Information Processing* **14** 121 – 138.
- Frantzeskakis, Linos F., Warren B. Powell. 1990. Successive linear approximation procedure for stochastic, dynamic vehicle allocation problems. *Transportation Science* **24** 40 – 57.
- Frost, M., B. Jorgenson, J. Kennington, A. Madhavan. 2008. Optimizing cash management for large scale bank operations. Technical Report 08-EMIS-03, Department of Engineering Management, Information, and Systems, Southern Methodist University, Dallas, TX 75275.
- Garey, M. R., David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, NY.
- Gen, M., R. Cheng. 2003. Evolutionary network design: Hybrid genetic algorithms approach. *International Journal of Computational Intelligence and Applications* **3** 357–380.
- Gen, M., R. Cheng, S.S. Oren. 2001. Network design techniques using adapted genetic algorithms. *Advances in Engineering Software* **32** 731 – 744.
- Gen, Mitsuo, Anup Kumar, Jong Ryul Kim. 2005. Recent network design techniques using evolutionary algorithms. *International Journal of Production Economics* **98** 251 – 261.

- Gintner, V., N. Kliewer, L. Suhl. 2005. Solving large multiple-depot multiple-vehicle-type bus scheduling problems in practice. *OR Spectrum* **27** 507–523.
- Glover, Fred. 1990. Tabu search: A tutorial. *Interfaces* **20** 74–94.
- Gottlieb, J., L. Paulmann. 1998. Genetic algorithms for the fixed charge transportation problem. *Proceedings of the IEEE Conference on Evolutionary Computation, Anchorage, AK, May 1998*. 330 – 335.
- Gray, Paul. 1971. Exact solution of the fixed-charge transportation problem. *Operations Research* **19** 1529–1583.
- Guo, Y., T. Mellouli, L. Suhl, M. Thiel. 2006. A partially integrated airline crew scheduling approach with time-dependent crew capacities and multiple home bases. *European Journal of Operational Research* **171** 1169–1181.
- Haberl, J. 1991. Exact algorithm for solving a special fixed-charge linear programming problem. *Journal of Optimization Theory and Applications* **69** 489–529.
- Haghani, Ali, Sei-Chang Oh. 1996. Formulation and solution of a multi-commodity, multi-modal network flow model for disaster relief operations. *Transportation Research, Part A: Policy and Practice* **30** 231 – 250.
- Herer, Y. T., M. J. Rosenblatt, I. Hefter. 1996. Fast algorithms for single-sink fixed charge transportation problems with applications to manufacturing and transportation. *Transportation Science* **30** 276 – 290.
- Hirsch, W. M., G. B. Dantzig. 1968. The fixed charge problem. *Naval Research Logistics Quarterly* **15** 413 – 424.
- Holland, J. H. 1975. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, Ann Arbor, MI.
- Jaramillo, J. H., J. Bhadury, R. Batta. 2002. On the use of genetic algorithms to solve location problems. *Computers and Operations Research* **29** 761 – 779.

- Jarrah, Ahmad I.Z., Gang Yu, Nirup Krishnamurthy, Ananda Rakshit. 1993. Decision support framework for airline flight cancellations and delays. *Transportation Science* **27** 266 – 280.
- Jo, Jung-Bok, Y. Li, M. Gen. 2007. Nonlinear fixed charge transportation problem by spanning tree-based genetic algorithm. *Computers and Industrial Engineering* **53** 290 – 298.
- Kennington, J. 1976. Fixed-charge transportation problem: A computational study with a branch-and-bound code. *AIIE Transactions* **8** 241 – 247.
- Kennington, J., E. Unger. 1976. A new branch-and-bound algorithm for the fixed-charge transportation problem. *Management Science* **22** 1116 – 1126.
- Kim, Dukwon, Panos M. Pardalos. 1999. A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. *Operations Research Letters* **24** 195 – 203.
- Kim, Hakjin, John Hooker. 2002. Solving fixed-charge network flow problems with a hybrid optimization and constraint programming approach. *Annals of Operations Research* **115** 95–124.
- Kliewer, Natalia, Taieb Mellouli, Leena Suhl. 2006. A time-space network based exact optimization model for multi-depot bus scheduling. *European Journal of Operational Research* **175** 1616 – 1627.
- Lamar, Bruce W., Chris A. Wallace. 1997. Revised-modified penalties for fixed charge transportation problems. *Management Science* **43** 1431 – 1436.
- Marín, Angel. 2006. Airport management: Taxi planning. *Annals of Operations Research* **143** 191–202.
- Marín, Angel, Esteve Codina. 2008. Network design: Taxi planning. *Annals of Operations Research* **157** 135 – 151.
- Michalewicz, Zbigniew, George A. Vignaux, Matthew Hobbs. 1991. Nonstandard genetic algorithm for the nonlinear transportation problem. *ORSA Journal on Computing* **3** 307 – 316.

- Murty, Katta G. 1968. Solving the fixed charge problem by ranking the extreme points. *Operations Research* **16** 268–279.
- Nahapetyan, A., P. M. Pardalos. 2008. Adaptive dynamic cost updating procedure for solving fixed charge network flow problems. *Computational Optimization and Applications* **39** 37–50.
- Ortega, F., L. Wolsey. 2003. A branch-and-cut algorithm for the single commodity uncapacitated fixed-charge network flow problem. *Networks* **41** 143 – 158.
- Palekar, Udatta S., Mark H. Karwan, Stanley Zionts. 1990. A branch-and-bound method for the fixed charge transportation problem. *Management Science* **36** 1092 – 1105.
- Powell, Warren B. 1986. Stochastic model of the dynamic vehicle allocation problem. *Transportation Science* **20** 117 – 129.
- Sheng, S., Z. Dachen. 2006. New genetic algorithm for the fixed charge transportation problem. *Proceedings of the World Congress on Intelligent Control and Automation, Dalian, China, June 2006*, vol. 2. 7039 – 7043.
- Sheng, S., Z. Dachen, X. Xiaofei. 2006. Genetic algorithms for the transportation problem with discontinuous piecewise linear cost function. *International Journal of Computer Science and Network Security* **6** 182–190.
- Sherali, Hanif D., Cihan H. Tuncbilek. 1997. Static and dynamic time-space strategic models and algorithms for multilevel rail-car fleet management. *Management Science* **43** 235 – 250.
- Steinberg, David I. 1970. The fixed charge problem. *Naval Research Logistics Quarterly* **17** 217–236.
- Sun, M., Patrick G. McKeown. 1993. Tabu search applied to the general fixed charge problem. *Annals of Operations Research* **41** 405 – 420.
- Sun, Minghe, Jay E. Aronson, Patrick G. McKeown, Dennis Drinka. 1998. Tabu search heuristic procedure for the fixed charge transportation problem. *European Journal of Operational Research* **106** 441 – 456.

- Thengvall, Benjamin G., Jonathan F. Bard, Gang Yu. 2003. A bundle algorithm approach for the aircraft schedule recovery problem during hub closures. *Transportation Science* **37** 392 – 407.
- Tohyama, H., K. Ida, C. Teramatsu. 2006. A proposal of a genetic algorithm for fixed-charge transportation problem and related numerical experiments. *Journal of Japan Industrial Management Association* **57** 222 – 230.
- Tomlin, J. 1971. An improved branch-and-bound method for integer programming. *Operations Research* **19** 1070–1074.
- Walker, W. E. 1976. A heuristic adjacent extreme point algorithm for the fixed charge problem. *Management Science* **22** 587–596.
- Wu, Peiling, Joseph C. Hartman, George R. Wilson. 2005. An integrated model and solution approach for fleet sizing with heterogeneous assets. *Transportation Science* **39** 87 – 103.
- Yan, S., C.-G. Lin. 1997. Airline scheduling for the temporary closure of airports. *Transportation Science* **31** 72–82.
- Yan, S., L. Shih. 2007. A time-space network model for work team scheduling after a major disaster. *Journal of the Chinese Institute of Engineers* **30** 63 – 75.
- Yang, L., Y. Feng. 2007. A bicriteria solid transportation problem with fixed-charge under stochastic environment. *Applied Mathematical Modeling* **31** 2668–2683.
- Zawack, Daniel J., Gerald L. Thompson. 1987. Dynamic space-time network flow model for city traffic congestion. *Transportation Science* **21** 153 – 162.

Table 1: Problem Characteristics

Set	Problem Type	Range of $R_{ir}$	Range of $c_{ijrs}$	Range of $f_{ijrs}$
1	LLL	10-20	0-10	200-600
2	LLM	10-20	0-10	2,000-6,000
3	LLH	10-20	0-10	20,000-60,000
4	LML	10-20	10-100	200-600
5	LMM	10-20	10-100	2,000-6,000
6	LMH	10-20	10-100	20,000-60,000
7	LHL	10-20	100-1,000	200-600
8	LHM	10-20	100-1,000	2,000-6,000
9	LHH	10-20	100-1,000	20,000-60,000
10	MLL	100-200	0-10	200-600
11	MLM	100-200	0-10	2,000-6,000
12	MLH	100-200	0-10	20,000-60,000
13	MML	100-200	10-100	200-600
14	MMM	100-200	10-100	2,000-6,000
15	MMH	100-200	10-100	20,000-60,000
16	MHL	100-200	100-1,000	200-600
17	MHM	100-200	100-1,000	2,000-6,000
18	MHH	100-200	100-1,000	20,000-60,000
19	HLL	1,000-2,000	0-10	200-600
20	HLM	1,000-2,000	0-10	2,000-6,000
21	HLH	1,000-2,000	0-10	20,000-60,000
22	HML	1,000-2,000	10-100	200-600
23	HMM	1,000-2,000	10-100	2,000-6,000
24	HMH	1,000-2,000	10-100	20,000-60,000
25	HHL	1,000-2,000	100-1,000	200-600
26	HHM	1,000-2,000	100-1,000	2,000-6,000
27	HHH	1,000-2,000	100-1,000	20,000-60,000

Table 2: Summary of Solution Results for Class 1 Problems

Set	Problem Class	Solved to optimality by both methods	M1		M2		M1/M2						
			Percent solved to optimality (000,000s)	Average number of Pivots (000,000s)	Average number of B&B Nodes (000s)	Average CPU Time HH:MM:ss	Percent solved to optimality (000,000s)	Average number of Pivots (000,000s)	Average number of B&B Nodes (000s)	Average CPU Time HH:MM:ss	Average Pivots Ratio	Average B&B Nodes Ratio	Average CPU Time Ratio
1	LLL	23	77%	27.6	979.2	0.36:57	93%	4.0	184.1	0:03:41	9.0	6.4	12.0
2	LLM	22	73%	15.5	703.8	0:16:24	97%	3.1	213.1	0:02:54	6.0	4.8	6.5
3	LLH	25	83%	37.9	2,017.5	0:38:21	97%	7.6	468.6	0:06:58	7.0	5.8	7.5
4	LML	30	100%	0.6	18.0	0:00:41	100%	0.2	8.0	0:00:11	2.4	1.9	2.2
5	LMM	22	73%	15.7	528.2	0:21:28	90%	3.1	161.3	0:02:52	5.4	3.9	7.3
6	LMH	23	77%	28.8	1,249.5	0:29:20	97%	4.7	323.8	0:04:17	7.6	4.7	8.2
7	LHL	30	100%	0.0	0.1	0:00:00	100%	0.0	0.0	0:00:00	1.2	2.3	1.0
8	LHM	30	100%	0.5	16.6	0:00:35	100%	0.2	8.1	0:00:13	2.4	2.0	2.0
9	LHH	23	77%	21.9	694.8	0:31:26	100%	3.5	176.9	0:03:19	7.8	4.9	9.6
10	MLL	29	97%	3.3	104.1	0:04:54	100%	1.0	37.7	0:01:16	3.1	2.8	3.2
11	MLM	9	30%	47.1	1,720.2	1:08:23	87%	4.5	231.7	0:04:50	12.0	7.9	16.8
12	MLH	6	20%	25.8	1,238.1	0:34:19	53%	4.3	252.1	0:04:21	12.8	10.5	19.1
13	MML	30	100%	0.0	0.2	0:00:00	100%	0.0	0.1	0:00:00	1.6	2.9	1.2
14	MMM	29	97%	8.6	253.8	0:13:44	100%	2.1	67.8	0:02:57	3.1	2.8	3.0
15	MMH	7	23%	28.4	909.1	0:41:36	60%	2.8	119.1	0:03:00	14.5	9.7	19.5
16	MHL	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.1	1.5	0.9
17	MHM	30	100%	0.0	0.1	0:00:00	100%	0.0	0.1	0:00:00	1.5	2.0	1.1
18	MHH	28	93%	13.0	364.9	0:22:17	100%	2.3	75.7	0:03:18	4.3	3.5	5.5
19	HLL	30	100%	0.0	0.2	0:00:00	100%	0.0	0.1	0:00:00	1.5	2.3	1.2
20	HLM	29	97%	1.8	45.5	0:03:17	100%	0.9	34.7	0:01:16	2.1	1.9	2.2
21	HLH	10	33%	26.1	584.6	0:57:08	63%	31.7	1,150.2	0:35:20	2.6	1.3	4.7
22	HML	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.0	1.6	0.9
23	HMM	29	97%	0.0	0.1	0:00:00	100%	0.0	0.1	0:00:00	1.3	2.5	1.1
24	HMH	28	93%	3.8	90.3	0:06:53	97%	2.1	65.2	0:03:20	1.9	1.6	2.0
25	HHL	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.0	0.0	1.1
26	HHM	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.0	0.6	1.0
27	HHH	29	97%	0.0	0.1	0:00:00	100%	0.0	0.0	0:00:00	1.3	1.7	1.1

Table 3: Summary of Solution Results for Class 2 Problems

Set	Problem Class	Solved to optimality by both methods	M1			M2			M1/ M2				
			Percent solved to optimality (000,000s)	Average number of Pivots (000s)	Average CPU Time HH:MM:ss	Percent solved to optimality (000,000s)	Average number of Pivots (000,000s)	Average number of B&B Nodes (000s)	Average Pivots Ratio	Average B&B Nodes Ratio	Average CPU Time Ratio		
1	LLL	14	47%	14.1	625.4	0:27:11	67%	2.4	115.6	0:03:53	7.9	4.0	8.6
2	LLM	14	47%	29.8	1,616.5	0:53:09	73%	6.5	423.3	0:09:34	5.8	4.3	7.1
3	LLH	10	33%	19.1	1,058.6	0:36:19	60%	2.9	194.3	0:04:15	6.0	4.2	6.8
4	LML	23	77%	4.8	152.9	0:08:29	100%	2.6	102.5	0:03:50	2.9	2.1	3.2
5	LMM	19	63%	23.8	595.6	0:53:17	80%	4.1	170.3	0:06:40	6.5	4.6	8.2
6	LMH	13	43%	14.5	635.3	0:26:04	63%	3.5	218.3	0:04:58	5.8	3.7	7.1
7	LHL	29	97%	0.1	2.9	0:00:10	100%	0.0	1.3	0:00:02	1.6	2.0	1.4
8	LHM	29	97%	5.7	180.6	0:10:29	100%	2.8	103.9	0:04:08	2.7	2.1	2.6
9	LHH	17	57%	16.2	454.1	0:36:20	73%	3.4	140.2	0:05:38	6.6	4.1	8.8
10	MLL	12	40%	22.1	764.5	0:47:33	70%	5.2	207.7	0:08:45	5.3	3.8	5.5
11	MLM	1	3%	2.7	43.3	0:07:33	30%	2.0	59.6	0:04:08	1.3	0.7	1.8
12	MLH	5	17%	33.0	501.7	1:32:00	27%	7.6	261.6	0:13:23	4.2	2.1	6.1
13	MML	30	100%	1.8	29.1	0:04:08	100%	0.5	10.6	0:01:06	2.5	2.7	2.1
14	MMM	10	43%	11.6	411.0	0:25:38	60%	2.8	93.2	0:05:56	5.5	4.0	5.7
15	MMH	4	13%	2.2	41.3	0:05:51	33%	0.6	15.3	0:01:07	2.4	2.8	2.3
16	MHL	30	100%	0.0	0.1	0:00:00	100%	0.0	0.1	0:00:00	1.2	0.7	1.1
17	MHM	27	100%	0.0	3.2	0:00:03	100%	0.0	1.2	0:00:02	2.5	5.1	1.7
18	MHH	13	43%	8.3	236.0	0:20:24	63%	1.1	43.8	0:02:27	13.9	9.8	19.6
19	HLL	30	100%	0.0	1.7	0:00:06	100%	0.0	1.6	0:00:05	1.4	1.4	1.2
20	HLM	14	47%	14.5	200.5	0:42:55	70%	5.0	151.9	0:12:11	2.5	1.3	3.1
21	HLH	1	3%	1.5	45.2	0:04:54	13%	1.4	131.9	0:02:35	1.1	0.3	1.9
22	HML	30	100%	0.0	0.1	0:00:00	100%	0.0	0.0	0:00:00	1.2	1.0	1.2
23	HMM	30	100%	3.1	100.5	0:06:31	100%	1.0	46.3	0:01:26	2.0	3.4	1.5
24	HMH	11	37%	7.6	104.0	0:22:22	60%	2.5	70.1	0:06:01	2.8	1.3	3.6
25	HHL	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.1	0.1	1.1
26	HHM	30	100%	0.0	0.0	0:00:00	100%	0.0	0.0	0:00:00	1.2	0.6	1.2
27	HHH	30	100%	0.0	3.0	0:00:06	100%	0.0	1.5	0:00:04	2.2	3.3	1.5

Table 4: The Normal Distributions

Requirements	For Uniform Distribution		For Normal Distribution	
	<i>minrhs</i>	<i>maxrhs</i>	mean	stdev
L	10	20	15	3.04
M	100	200	150	30.40
H	1000	2000	1500	303.95

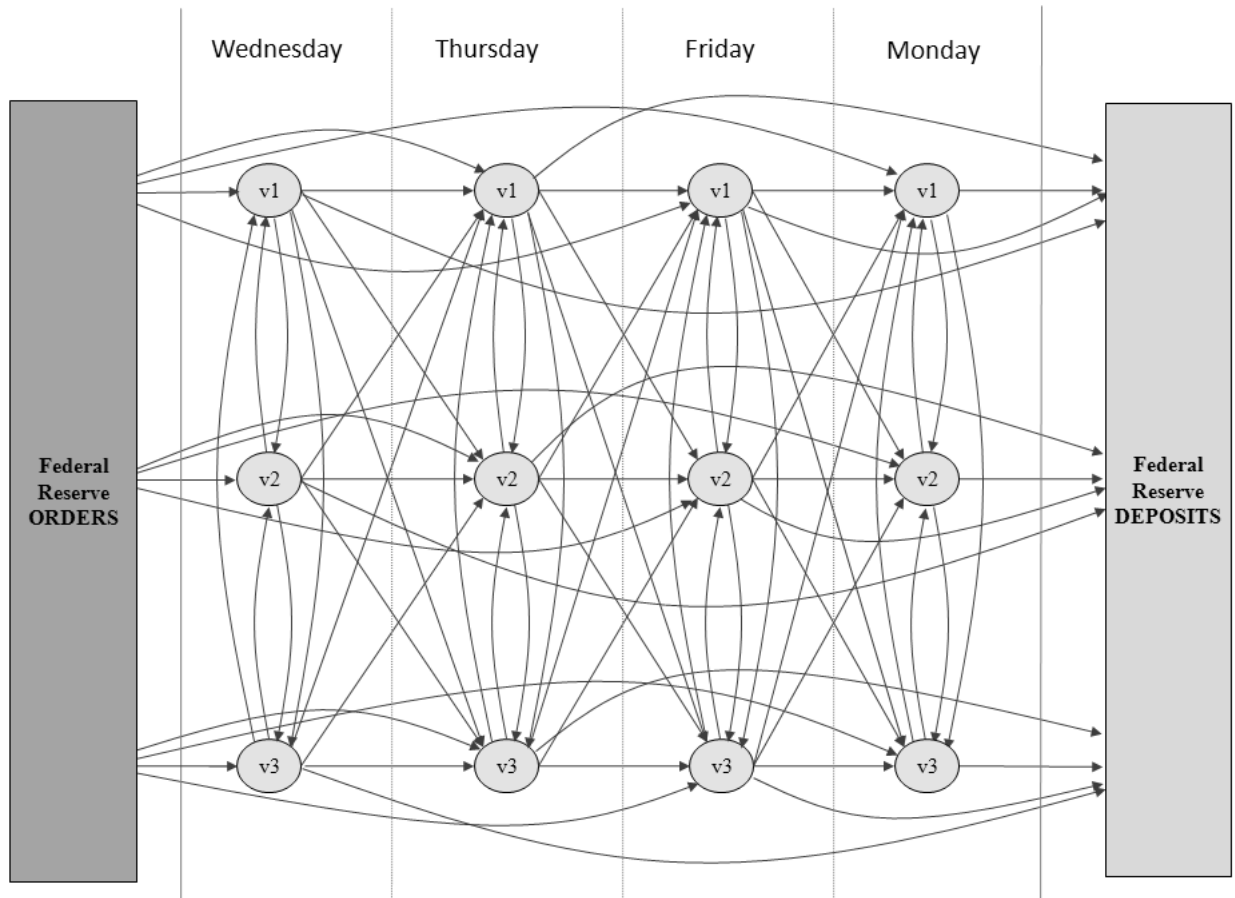
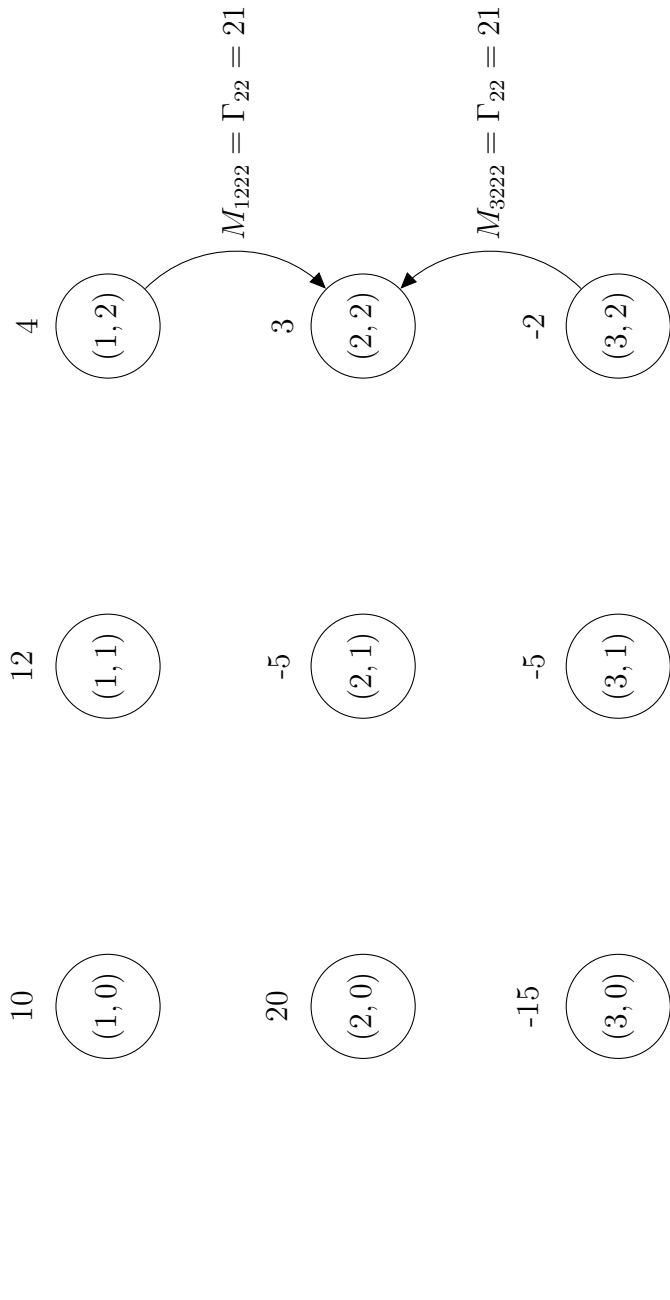


Figure 1: Time-Space Network Representation of Bank Problem.

Table 5: Results for Test Problems Using the Normal Distribution

Problem Type	Problem Number	M1	M2	M1/M2
		CPU Time HH:MI:SS.xx	CPU Time HH:MI:SS.xx	CPU Time Ratio
LLL	P400	0:11:50.42	0:00:53.20	13.4
LLL	P700	4:00:15.70*	0:34:52.13	-
LLL	P1100	4:00:16.30*	4:00:39.80*	-
LLL	P1600	0:31:37.81	0:03:54.62	8.1
LLM	P1600	0:02:17.99	0:00:31.83	4.3
LLM	P1700	1:46:48.74	0:08:12.10	13.0
LLH	P1300	1:15:48.85	0:26:36.42	2.8
LLH	P1400	2:03:22.35	0:44:36.57	2.8
LLH	P1600	0:02:44.67	0:00:28.16	5.8
LLH	P2700	0:07:57.73	0:03:11.31	2.5
LMM	P700	4:00:12.60*	0:56:16.87	-
LMM	P1100	4:00:32.20*	4:00:14.20*	-
LMM	P1700	0:03:24.08	0:00:33.29	6.1
LMH	P1300	1:52:24.66	0:22:27.51	5.0
LMH	P1600	0:01:35.47	0:00:20.19	4.7
LMH	P1700	2:18:49.48	0:05:10.73	26.8
LMH	P2700	0:19:05.36	0:06:26.31	3.0
LHH	P1100	4:00:17.00*	4:00:33.80*	-
LHH	P1700	0:04:16.28	0:00:44.60	5.7
LHH	P2000	4:00:16.00	4:00:26.60*	-
MLM	P1000	4:00:08.00	1:38:57.31	-
MLM	P1100	4:00:13.10*	4:00:23.40*	-
MLM	P1300	4:00:09.90*	4:00:21.60*	-
MLM	P1400	4:00:20.10*	4:00:20.00*	-
MLM	P2800	4:00:17.60*	1:22:43.09	-
MLH	P1200	4:00:39.50*	4:00:53.40*	-
MLH	P2200	2:01:26.31	0:16:48.07	7.2
MLH	P2300	0:11:59.99	0:02:24.41	5.0
MMM	P1100	2:03:46.04	0:19:16.41	6.4
MMH	P1000	4:00:09.90*	0:40:31.81	-
MMH	P2200	4:00:13.90*	0:25:17.20	-
MMH	P2600	4:00:07.60*	1:47:26.50	-
MHH	P900	0:01:02.25	0:00:26.38	2.4
MHH	P1100	1:41:21.44	0:20:32.71	4.9
HLH	P1500	4:00:09.60*	4:00:31.60*	-
HLH	P1900	4:00:10.30*	4:00:45.20*	-

\*terminated due to 4 hour time limit



**Excess Supply:** 15      2

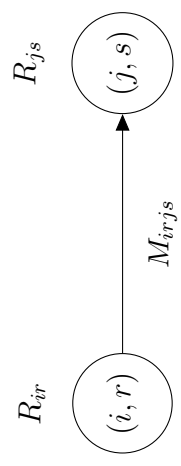


Figure 2: Example of  $\Gamma_{22}$  Calculation

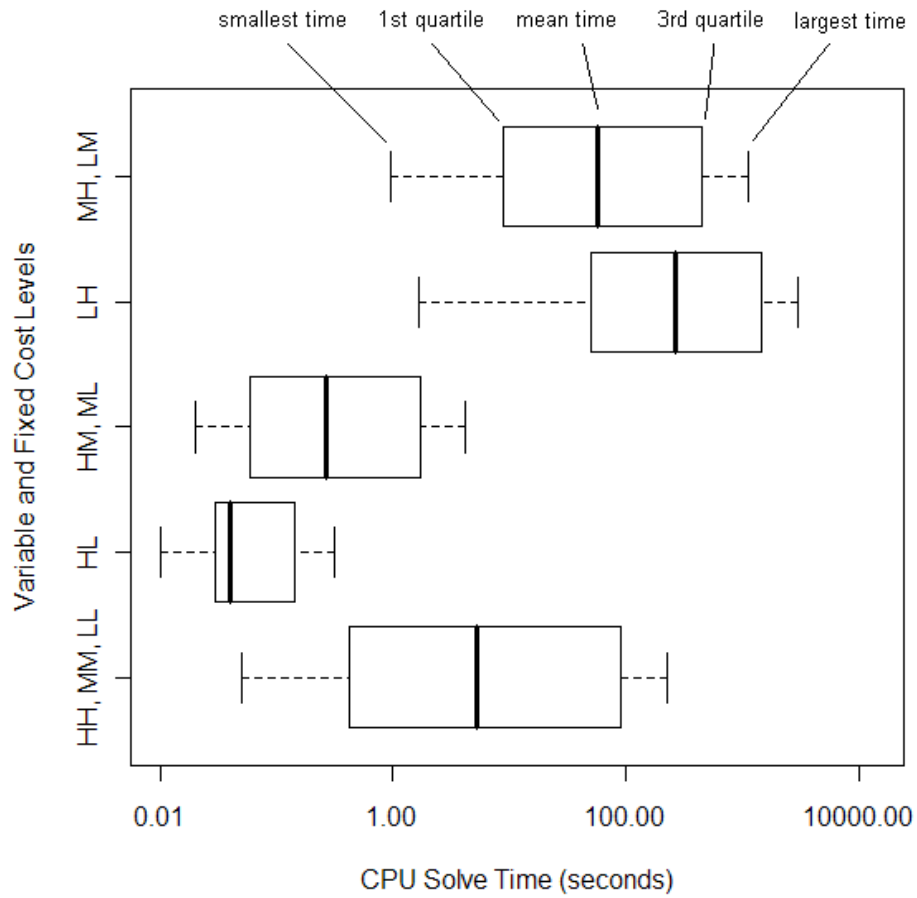


Figure 3: Solve Time versus Variable and Fixed Cost Levels (HL implies a high level of variable cost and a low level of fixed cost)

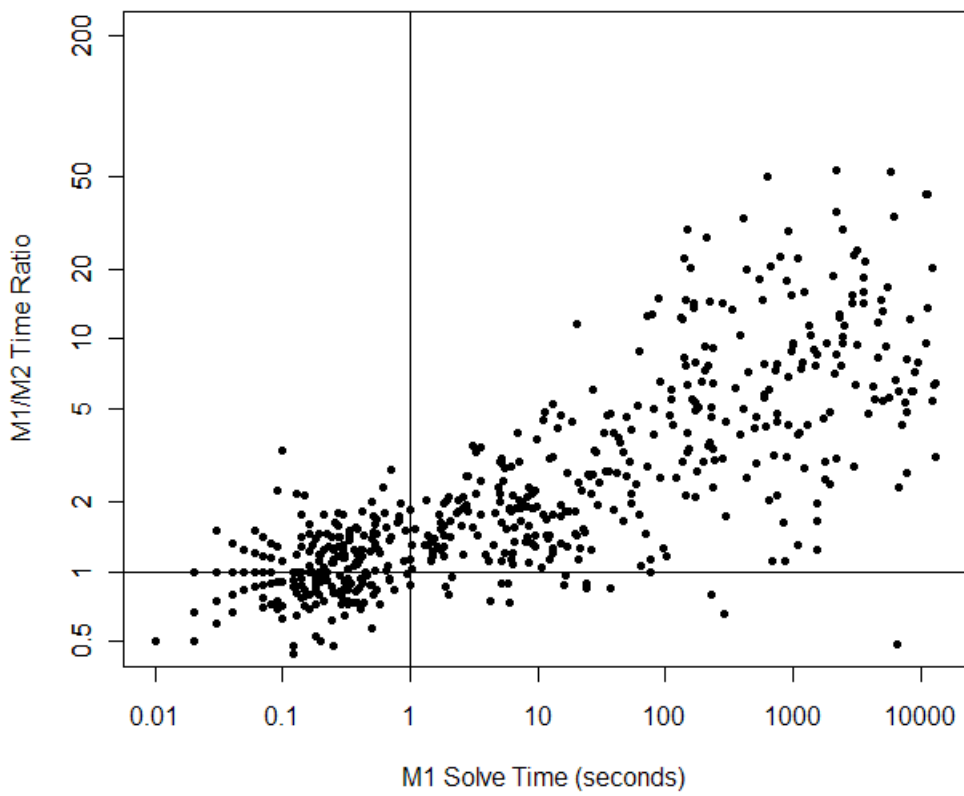


Figure 4: Log-log Graph of Class 1 Solve Time Ratio versus Problem Difficulty

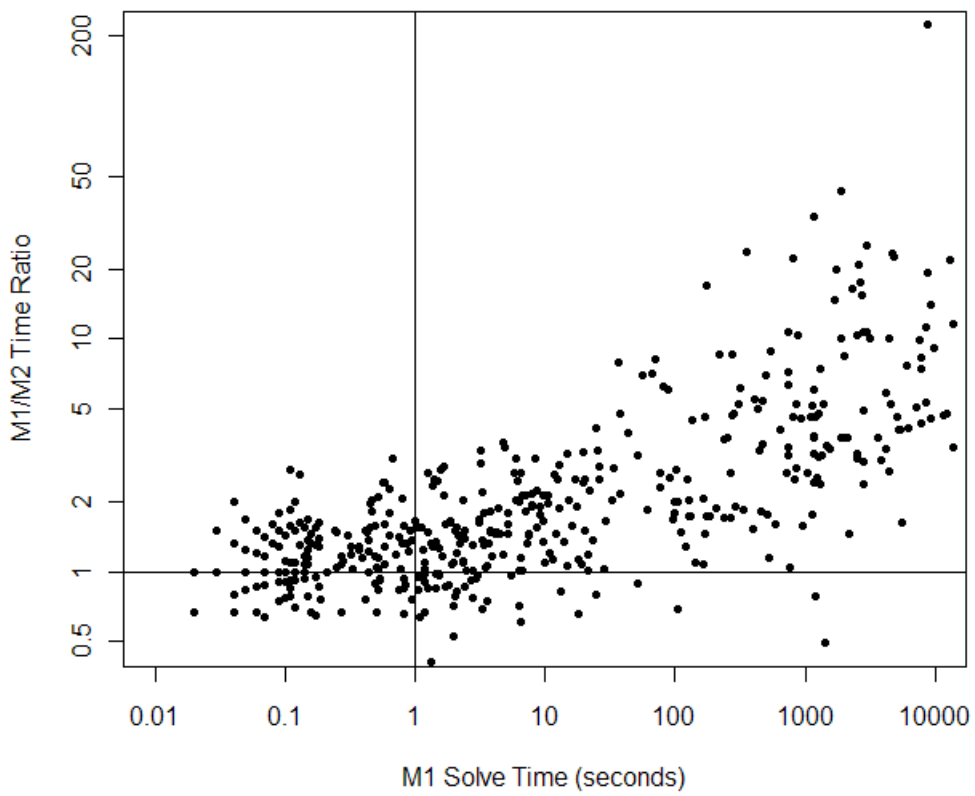


Figure 5: Log-log Graph of Class 2 Solve Time Ratio versus Problem Difficulty