HARNESSING CERTAINTY TO SPEED TASK-ALLOCATION ALGORITHMS FOR MULTI-ROBOT SYSTEMS

An Undergraduate Research Scholars Thesis

by

DENISE IRVIN

Submitted to the Undergraduate Research Scholars program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Shell

May 2017

Major: Computer Science

TABLE OF CONTENTS

ABSTRACT

Harnessing Certainty to Speed Task-Allocation Algorithms for Multi-Robot Systems

Denise Irvin
Department of Computer Science and Engineering
Texas A&M University

Research Advisor: Dr. Shell
Department of Computer Science and Engineering
Texas A&M University

Some problems are best solved by systems of multiple robots, in which each robot is assigned one task. A multi-robot system can, upon the start of a series of tasks, compute the optimal task allocation for best performance of the team. For certain systems, during runtime, changes in the environment, tasks, and state of individual robots might change which allocation of tasks to robots is optimal, and the performance of the team would improve if the robots switched tasks. Because communication between robots is expensive, in some cases it is better to calculate an interval in which changes in the environment, tasks, and robots are not significant enough to render the original allocation suboptimal. This way, robots only initiate communication and correction if the system is likely to switch tasks, which limits the costs of communication and computation in the system. In the problem of task allocation of single robot, single task cases where environments and thus optimal assignments are expected to vary over time, some knowledge of the system might help reduce computation and make possible a more scalable algorithm for determining cost changes. In some systems, some costs may be known not to vary over time. This research proposes creating and analyzing cost matrices of assignments to examine if taking advantage of the certainty of some variables will improve performance. If

successful, models for exploiting certainty of task allocation will take less computation than calculating ranges for all variables, and will save resources during runtime.

ACKNOWLEDGMENTS

I want to thank Dr. Shell for the help and support he gave me while exploring this topic. It has been a pleasure working with him.

NOMENCLATURE

SR	Single Robot
ST	Single Task
MRTA	Multiple Robot Task Allocation
OAP	Optimal Assignment Problem
C	Cost matrix representing the costs of assigning a particular robot to a particular task; also called the Objective Function Coefficients
$\underline{C}$	The matrix containing the lower boundaries of the cost matrix c
$\overline{C}$	The matrix containing the upper boundaries of the cost matrix c
X^*	The optimal assignment matrix for a OAP problem
$\theta(X^*)$	A set containing all matrices c which for which X^* is optimal
V_v	Set of indices (i,j) where $\underline{c}_{ij} < \overline{c}_{ij}$
V_c	Set of indices (i,j) where $\underline{c}_{ij} = \overline{c}_{ij}$

CHAPTER I

INTRODUCTION

Research Question

The multi-robot task allocation (MRTA) problem assigns tasks to robots while minimizing a cost function, usually time, of a system completing all its tasks. This problem can be generalized into an Optimal Assignment Problem (OAP), as described by Siciliano and Khatib [1], and linear programming methods can be applied. Specifically, in this paper we consider the single robot, single task, instantaneous assignment problem, SR-ST-IA, meaning we are assigning exactly one task to exactly one robot without planning for future task allocation. For a system with n robots and m tasks, an $n \times m$ matrix cost matrix, C is typically given for which each value c_{ij} is the cost associated with assigning the i-th robot to the j-th task. Without loss of generality, we assume n = m, because dummy robots or extra tasks could be inserted added to the system. The optimal assignment matrix, called X^*, is an $n \times m$ matrix where $x_{ij} = 1$ if the i-th robot is assigned to the j-th task, and $x_{ij} = 0$ otherwise. Mathematically, this problem is defined by equations 1-5.

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} \tag{1}$$

subject to

$$\sum_{j=1}^{m} x_{ij} = 1 \ \forall i, \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \ \forall \mathrm{j}, \tag{3}$$

$$0 \leq x_{ij} \leq 1 \ \{i, j\}, \tag{4}$$

$$x_{ij} \in \mathbb{Z}^{+} \ \{i, j\}. \tag{5}$$

We want to compute ranges uncertain costs can deviate without breaking optimality. This range of costs lie in an n^2-polytope called $\theta(X^*)$. Ideas from sensitivity analysis, (SA), discussed at length in by Ward and Wendell [2] and Gal [3], analyzes the effects individual variables have on the whole system and can be applied to create ranges for each individual assignment cost. This problem often has multiple feasible solutions, where the assignment variables in X are partitioned into basic and nonbasic variables [4]. Basic variables are variables that correspond to a vector, for a feasible basis. In the equation below, J_k is an array of the indices corresponding to basic variables, N_k is an array of indices corresponding to nonbasic variables, B_k is a set of columns of the constraint matrix corresponding to basic variables and A_{N_k} is a set of columns of the constraint matrix corresponding to nonbasic variables. A critical region, R_k, where k is the index of a feasible solution, is the area enclosed by the constraints of the basic variable set, where optimality is guaranteed to be preserved, defined by equation 6,

$$R_k = \left\{ C \in \mathbb{R}^{n^2} : C_{N_k} - C_{J_k} B_k^{-1} A_{N_k} \geq 0 \right\}. \tag{6}$$

Lin and Wen discuss the degeneracy of the OAP [5], and that there may be more than one basis corresponding to an optimal solution, so we must consider the union of all critical regions, described by equations 7-8.

$$\theta(X^*) = \bigcup_{k \in H} R_k \tag{7}$$

Where:

$$H = \{ k : X^*_{J_k} = B_k^{-1}, X_{N_k}^* = 0 \} \tag{8}$$

In some situations, it is unreasonable or less useful to model the costs as static, scalar values, because the costs are uncertain or expected to change over time. In cases where we a priori know the bounds in which costs fluctuate, we can model bounds with matrices $\overline{C}$, which contains the upper limit of the costs of each assignment and $\underline{C}$, which contains the lower limit of the costs of each assignment. In our experimentation, we run tests where costs have an upper and lower bound equal to each other, effectively fixing the cost and where costs have an upper and lower bound of infinity and negative infinity, respectively. We introduce the notation V_v as the set of indices (i, j) for each c_{ij} where $\underline{c}_{ij} < \overline{c}_{ij}$. We define V_c as the set of indices (i, j) for each c_{ij} where $\underline{c}_{ij} = \overline{c}_{ij}$.

For models with variable costs over time, the performance of the system can potentially be improved by changing which robots are assigned to which tasks. An algorithm that calculates the potential utility of a change in task allocation must either periodically poll each agent and compute an optimal assignment, requiring expensive communication, or be triggered by a significant change in cost of one of robot-task pairs. This is done by assigning the initial optimal assignment, and then calculating the range of cost values an assignment can fluctuate while the current allocation is guaranteed to be optimal [2]. If a range is violated, then the optimal assignment algorithm is recomputed. These calculations are costly, and may be simplified in systems where some of the costs are known to never fluctuate over time.

We consider four categories of certainty. The first category is complete certainty, where no costs in the model are uncertain. The last category is complete uncertainty, in which all costs are uncertain. Other systems possess mixed certainty, where some costs are known to be certain

and others expected to vary over time. We suggest breaking mixed certainty problems into two categories: structured mixed certainty and unstructured mixed certainty. For a cost matrix where each value is the cost of a particular robot performing a particular task, this paper identifies two kinds of structured mixed certainty. The first case is where the cost of every task for a particular robot is certain, and other robots in the system have uncertain values. In the $n \times m$ cost matrix, where n is the number of robots and m is the number of tasks, in the first case of certainty, the rows representing robots with fixed costs have certain values. The second case is of structured certainty is a system where the cost of performing a particular task is certain, regardless of the robot performing it. A matrix representing this situation would be composed of whole columns with fixed values and whole columns of ranged values. A case of unstructured certainty is a system where some robot-task assignment costs are certain, but for a particular task the cost is not certain for every robot and for a particular robot the cost for every task is not certain. In this cost matrix, there is no guaranteed pattern to which costs are certain and which costs are uncertain.

This paper builds upon research done in uncertain cost systems, particularly in work done using ideas from sensitivity analysis like Ward and Wendell [2], Lin and Wen [5], and Nam and Shell [6] to explore unstructured certainty, beginning with cases with relatively few uncertain costs.

Problem Example

Let us consider an example case of unstructured certainty, shown in Fig. 1 and Table 1, where only one cost value is uncertain. Say we have a terrain with destinations $D_{1,2}$ and need

exactly one of our two robots $R_{1,2}$ to report at each destination. The cost of assigning a robot to a task is measured by the time it takes a robot R_i to reach a destination D_j. Because we assume each robot is taking the shortest path and going at the fastest speed it can, all the costs are known. Suppose there is a bridge between R_1 and D_1 that may be impassable, rendering the cost associated for this assignment unknown, but within the bounds of 10, the number of minutes R_1 takes to reach D_1 when it can take the bridge, and 60 is the number of minutes it takes when the bridge cannot be used.

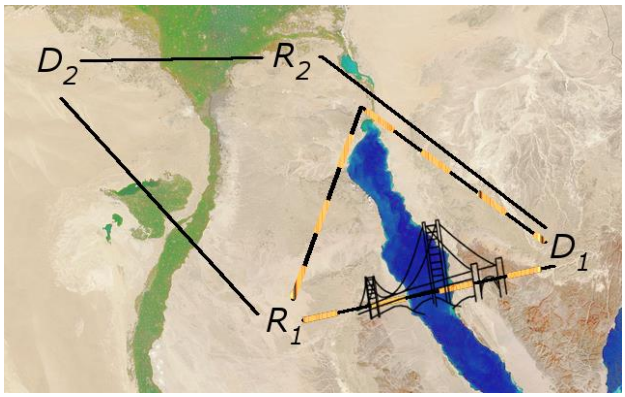


Fig 1. Navigation Example Image. We have two robots $R_{1,2}$ and two destinations $D_{1,2}$, and our goal is to have a robot in each destination in the shortest possible time.

Table 1. Navigation Example Cost Matrix (C).

	D_1	D_2
R_1	[10,60]	[20,20]
R_2	[30,30]	[10,10]

The initial assignment of this problem is $X_0^* = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, if the cost used for the assignment of R_1 to D_1 is the average of the two possible costs. Intervals can then be calculated

for how much the uncertain cost can change, before the optimality is violated and the task will be performed faster if the assignments are switched. For this case, the threshold cost for the assignment $c_{11} = 40$. If a few minutes into execution, R_1 discovers the bridge to be impassable, and recalculates the cost of execution to be $c_{11} = 60$, the tasks will then be reassigned so $X_1^* = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. We are interested in determining how we can simplify the computation of intervals where optimality is not violated when only some values are uncertain.

CHAPTER II

RELATED WORK

Sensitivity analysis, the field that analyzes how deviation of one or more values will affect other values in a problem, has inspired techniques regarding how systems allocate tasks. In 1990, Ward and Wendell [2] gave an overview of different approaches to sensitivity analysis. The last method considered in the paper, the tolerance approach, is designed for multiple deviations in variables. The maximum acceptable amount of deviation, or tolerance, is calculated for each variable in the problem. This limit on deviation, or tolerance, is multiplied by 100% to become the maximum tolerance percentage. This provides us with a basis to see how the percent change in a variable changes the final solution, as well as affecting other variables. The original tolerance method computes the overall tolerance, that is how much variables can simultaneously perturbed, based on the objective function. A weakness of the tolerance method is that if the optimal solution has alternate or near alternate optimal bases, the tolerance will be very small and unable to find these other bases. A weakness of this solution is that, in many nontrivial cases, the tolerance is close to zero. This approach has been extended to compute tolerances for individual variables. Later, Filippi [7] proposed a geometric algorithm to improve individual tolerances, with maximal ranges. These tolerance ranges are "safe," meaning that a cost can fluctuate in this range without breaking optimality, regardless of how other variables fluctuate.

Munkres [8] improved the Hungarian algorithm [9], a one time task-allocation assignment algorithm with a graph-based approach that assumes all costs are stable scalar values. In $\boldsymbol{O(n^3)}$ time, the Hungarian algorithm finds the optimal assignment of a system. Shell and Liu

[4] modify this algorithm to create the Interval Hungarian Algorithm, which given an optimal assignment problem, computes the ranges each cost variable can independently deviate before optimality is lost. However, in a situation where robots keep track of how the cost of their assignment varies over time, this interval is not a safe interval, because other costs could change, and render the assignment suboptimal. In systems for which multiple costs are expected to change, such ranges offer little guidance.

Nam and Shell [6] suggest breaking up the group of robots into cliques, and invoking global communication when the costs of the cliques violates its safe interval. Robots then can report costs locally and thus consider a tolerance range in a lower dimension that must be broken before resorting to global communication (and computation).

In the same paper, the authors present an approximation algorithm for constructing $\boldsymbol{\theta}(\boldsymbol{X}^*)$ that only takes the union of a variable number of critical regions. Because a critical region is calculated from a basic variable set, the number of basic variable sets determine the number of critical regions. There are $\boldsymbol{n+m-1}$ basic variables in a basic variable set for the assignment problem [1]. From equation 8 we see that all basic variable sets contain the $\boldsymbol{n}$ variables where $\boldsymbol{x_{ij}=1}$ in the original assignment, as well as $\boldsymbol{n-1}$ additional variables. These remaining $\boldsymbol{n-1}$ variables are chosen from the $\boldsymbol{n^2-n}$ variables for which $\boldsymbol{x_{ij}=0}$, resulting in a possible $\binom{\boldsymbol{n^2-n}}{\boldsymbol{n-1}}$ critical regions. However, as seen in equation 8, $\boldsymbol{B_k}$ must be invertible, which only a subset of basic variable sets satisfy. Therefore $\binom{\boldsymbol{n^2-n}}{\boldsymbol{n-1}}$ is an upper bound on the number of critical regions, growing proportionally to $\boldsymbol{n}!$. As a result, there is significant performance improvement in an estimation choosing fewer critical regions and yielding high quality results in problems with

complete uncertainty. These estimates are acceptable approximations of $\boldsymbol{\theta}(\boldsymbol{X}^*)$ because critical regions may overlap. We conduct an experiment based on this idea in the estimation algorithm section. Unfortunately, because $\binom{\boldsymbol{n^2-n}}{\boldsymbol{n-1}}$ must be calculated when creating critical regions, computations become intractable after $\boldsymbol{n = 6}$ so experimentation is limited to smaller values of $\boldsymbol{n}$.

Their technique assumes every cost in the system can fluctuate, which possibly introduces unnecessary computation costs if there are some costs known to be static. In such cases, the system exhibits "mixed certainty," a property that describes systems with some costs that are certain and some costs uncertain. Inspired by these authors, we consider leveraging this a priori knowledge to simplify computation costs by widening intervals, first by removing redundant constraints and second by considering a heuristic to choose critical regions to include in a polynomial estimation algorithm for computing $\boldsymbol{\theta}(\boldsymbol{X}^*)$.

CHAPTER III

ALGORITHMS

Linear Constraint Removal

The n^2-polytope that defines the area a cost matrix C can fluctuate without violating the original optimal assignment can be described by a series of linear constraints. These constraints can be generated by considering basic variables and all critical regions. Each individual critical region is bounded by $(n-1)^2$ linear constraints, each equal to the C where each element is multiplied by a corresponding coefficient from a unimodular matrix. When all variable costs in a constraint have a zero coefficient, the linear constraint is trivial true, because all constraints are satisfied by the original assignment. Therefore, such constraints can be safely ignored. Depending on the size of n and the ratio of variable costs to fixed costs, there will be different ratios of trivial constraints to total constraints. Coefficients will never be zero for variables that are a part of the original assignment, so this reduction is only useful in certain cases.

Estimation Algorithm

The estimation algorithm modifies the computation of the exact method by only computing the bounds of some critical regions and taking their union, effectively choosing a smaller k in equation 7. The total number of critical regions grows factorially, while the number of boundaries for each region is $(n-1)^2$, although as we discussed some of these constraints may be trivial. Choosing a smaller number of critical regions therefore will cause a significant improvement in performance.

However, computation is saved only if critical regions can be chosen before enumeration. Currently, we enumerate critical regions in a random order. The goal of studying critical regions is to develop a heuristic that chooses critical regions to allow for better estimates of $\theta(X^*)$ with fewer enumerations than this randomized algorithm. As mentioned above, critical regions are calculated in equation 6 with a basic variable set, containing the n variables for which $x_{ij} = 1$ in the original assignment, as well as $n - 1$ additional variables. We tested heuristics that prioritized critical regions that chose elements of $|V_v|$ as the additional basic variables, elements of $|V_c|$, and ratios of the two but found no method that could consistently compete with a random ordering.

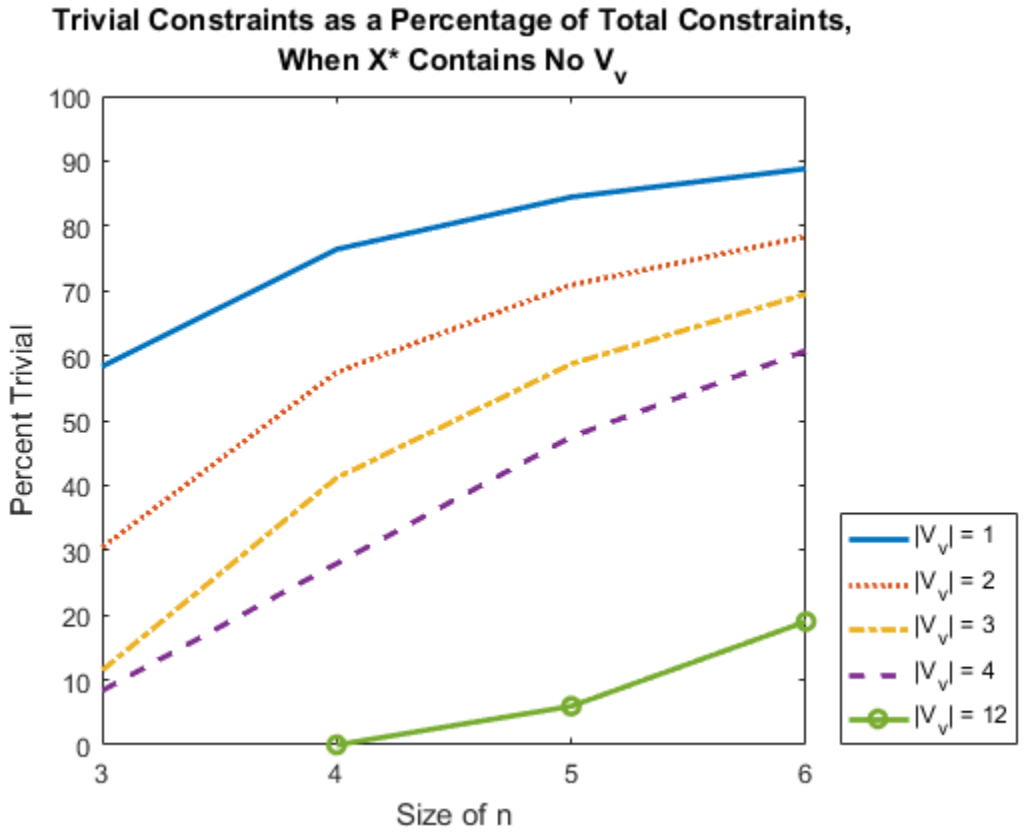
Upon analyzing critical regions with Monte Carlo, we found that, testing with $n = 3,4,5$ with varying sizes of $|V_v|$ that not all critical regions accept any of the tests, meaning none of the Monte Carlo tests appear in the area of a specific critical region. Generally, the number of "empty" regions varies with respect to $|V_v|$. Smaller $|V_v|$ values result in a smaller number of "nonempty" critical regions that contain many samples and larger $|V_v|$ values mean many nonempty critical region with less samples in each one. In fact, when $|V_v|$ is sufficiently small, i.e. $|V_v| \leq n$, there is a single critical region that contains every accepted sample. This pattern suggests that there is a general rule for which critical regions are needed to fully estimate the problem. We hope future research will uncover a heuristic that predicts the necessary critical regions for a given problem.

CHAPTER IV

RESULTS

Linear Constraint Removal Results

To analyze patterns in percentages of where linear constraints are trivial, experiments were ran with problems where $n = 3,4,5,6$. For each value of n, 10 trials where ran and averaged together. Separate trials where run where the number of unfixed costs, $|V_v|$, was equal to 1, 2, 3, 4 and 12 and the results are shown in Graph 1 below.



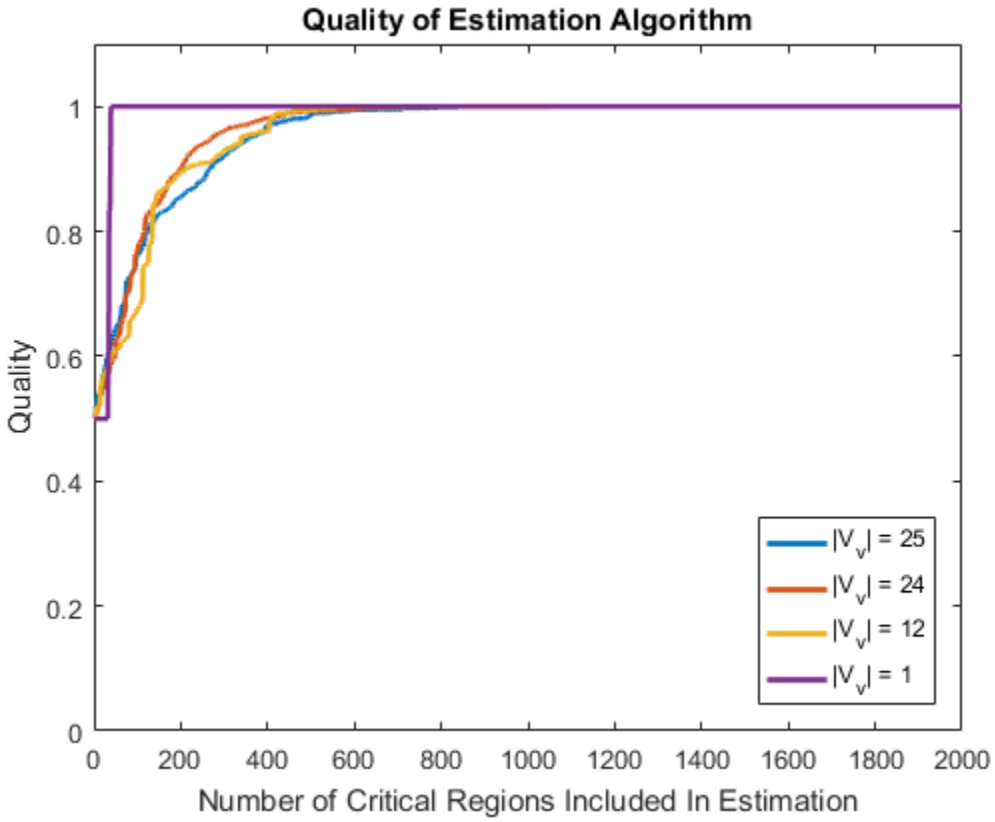
Graph 1 shows how the percent of trivial constraints vary with respect to changes in n and $|V_v|$.

As is expected, the more unfixed costs in the problem the fewer number of trivial constraints exist. The case $|V_v| = 12$ is of note because for $n = 5$, 48% or roughly half of the costs can vary. Removing trivial constraints will not improve the asymptotic run time of computing or testing if a point is in $\theta(X^*)$, runtime will be improved. This approach can be incorporated with the estimation algorithm to further speed computation.

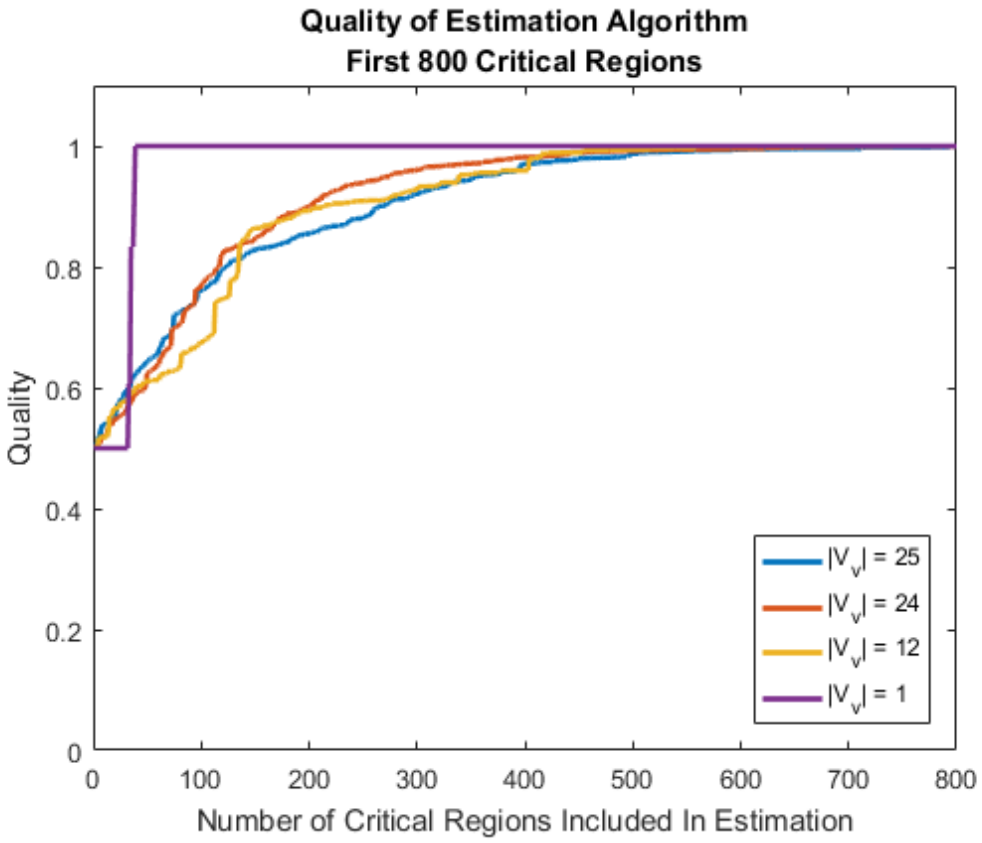
Estimation Algorithm Results

To test the quality of estimating $\theta(X^*)$ by only generating randomly selected critical regions, trials were performed on problems with size $n = 5$, for values of $|V_v| = 1,12,24,25$ which represent a problem that is 4%, 48%, 96% and 100% uncertain, respectively. Tests where ran by taking $100{,}000$ samples of randomly permutated matrices. A quality of 1 indicates that an estimation covers the whole area defined by $\theta(X^*)$. All results are shown in Graph 2 and

the first 800 regions of each test are shown in Graph 3 to more easily compare trials.



Graph 2 is the plot of the number of critical regions considered in the estimate against the how complete the estimate is of $\theta(X^*)$ for when $|V_v| = 1,12,24,25$. The line for $|V_v| = 25$ has 0% certainty and is therefore a baseline for comparison.



Graph 3 shows a larger view of the first 800 regions of the experiments of Graph 2.

From Graph 2, we see that in the different mixed certainty cases not all critical regions are needed to reach perfect quality and there is a potential for improvement in performance. This potential is most dramatic in the case $|V_v| = 1$, where less than 50 critical regions needed to be added together to reach full quality.

CHAPTER V

CONCLUSION

Mixed certainty task allocation problems have the opportunity for better performance than complete certainty problems. We analyzed trivial linear constraints, and how depending on the problem structure significant percentages of constraints can be removed. We also discussed an estimation algorithm that can compute the partial or complete $\theta(X^*)$ by choosing to union random critical regions. In the future, we will continue to research developing a heuristic for choosing necessary critical regions by exploiting the mixed certainty known for a given problem.

REFERENCES

[1] B. Siciliano and O. Khatib, Springer handbook of robotics. Springer, 2008.

[2] J. Ward and R. Wendell. "Approaches to sensitivity analysis in linear programming." Annals of Operations Research, vol. 27, pp. 3-38, 1990.

[3] T. Gal, *Postoptimal analyses parametric programming and related topics*. McGraw-Hill, 1979.

[4] L. Liu and D. Shell, "Assessing optimal assignment under uncertainty: An interval-based algorithm," Int. J. of Robotics Research, vol. 30, no. 7, pp. 936-953, 2011.

[5] C.-J Lin and U.-P Wen, "Sensitivity analysis of the optimal assignment," *Euro. J. of Operational Research,* vol. 149, pp.35-46, 2003.

[6] C. Nam and D. Shell, "When to do your own thing: Analysis of cost uncertainties in multi-robot task allocation at run-time," in Proc. of IEEE Int. Conf. on Robotics and Automation, 2015, pp. 1247-1254.

[7] C. Filippi, "A fresh view on the tolerance approach to sensitivity analysis in linear programming," Euro. J. of Operational Research, vol. 167, pp. 1-19, 2005.

[8] J. Munkres, "Algorithms for the assignment and transportation problems," Journal of the Soc. for Industrial and Applied Mathematics, vol. 5, no. 1, pp. 32-38, 1957.

[9] H. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, no. 1-2, pp. 83-97,1955.