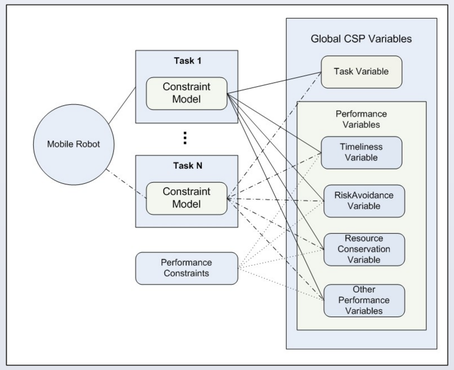
Constraint Programming Applications in Unmanned Aerial System Flight Pathing

Courtney Thurston, Dr. Richard S. Stansbury

Embry-Riddle Aeronautical University, Daytona Beach

Introduction

At the end of his 2007 PhD dissertation, Dr. Richard S. Stansbury expressed a need for future work on the research question he addressed: the modeling and evaluation of navigation and 'task' completion as a constraint programming problem. His research addressed only 2D topology, but the results showed strong potential for applications in 3D, too.[1] The goal of the research project I formulated in 2016—and have worked on since as the sole Principal Investigator—is to address this open question in the constraint programming field within computer science and computational math.



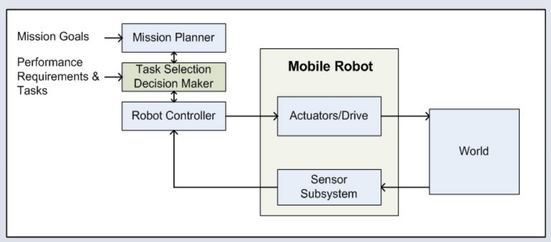
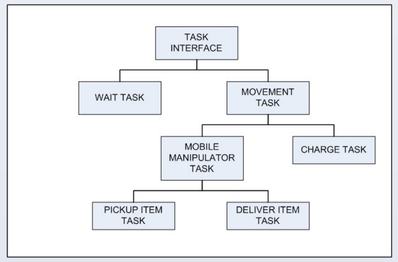


Figure 1 & 2: System and Mission Overview

The constraint satisfaction problem (CSP) is a search-based problem domain from the Artificial Intelligence community where the variables of the problem are assigned values that satisfy the constraints set upon them. Constraint programming techniques derived from constraint satisfaction research express very complex relationships between interdependent variables. CSPs have been applied to many application areas including scheduling, design, and image processing. However, they have not been utilized much for robotic applications due to the computational complexity (NP-Hard) of current solution algorithms.[7] These techniques allow the platform to make a multi-objective decision more quickly and with less resource-expenditure than rule-based systems can. A satisfying solution to the CSP model is defined as one such that the task selected will not only operate correctly but also satisfy the current requirements set forth upon the robot, and ideally exceed its performance expectations.[5]

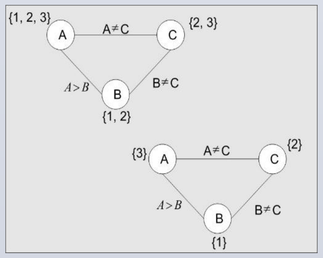
Methods and Current Work

System requirements may vary during operation to accommodate the robot's performance, environmental conditions, or user expectations. For instance, a mobile robot navigating through a congested environment should operate with greater caution rather than ensure timely delivery. The task selection requirements may be defined in terms of one or more performance objectives. Possible performance objectives may be the timeliness of the operation, risk avoidance, or resource conservation. These objectives (**Figure 3**, below) could conflict with one another under many scenarios; thus, it is a challenge to develop a decision-making system capable of selecting a task given multiple objectives. A task selection mechanism, such as the one presented in this research, can change requirements while continuing to ensure rationality.[2] In addition, the controller can change conflicting solutions when multiple requirements attempt to influence the decision.



The decision maker maintains a list of tasks, querying each task to determine if it is active—important to the mission. If a task is considered active, it requests the task's CSP. The task CSPs are merged into a single CSP that represents the entire problem. The CSP is solved, and its first N-solutions are extracted, where N is a runtime-defined number.[9] From these N-solutions, the best solution is selected and executed, modifying the flight plan and navigation plan when applicable.

Figure 4 below provides an example of a constraint satisfaction problem (upper-left) and its solution (lower-right). For each, a constraint graph is provided. The nodes represent the unsolved variables; the domain for each respective variable is presented in curly-braces next to the node. The arcs represent the constraints between the variables; the solution contains a single value in the domain for each variable.[9]



The decision-maker is constructed using many data structures and algorithms in theoretical and applied computer science. The dual-graph technique is used to compare binary and n^{th}-order CSPs. In dual graph, nodes represent constraints, and their domains are "the legal combinations of values between the variables within the constraint." The arcs between these constraint nodes represent common variables within the constraints and enforce that the variables' instantiation is the same for both constraints. In the dual-graph form, there is a new and legal binary CSP that may utilize any binary CSP solution or reduction technique.[8,9]

Traditionally, constraint propagation techniques are performed prior to solving the CSP with backtracking. Forward checking is a technique that can be utilized such that arc consistency is performed during the search; back-jumping makes the act of backtracking more intelligent within the CSP solver. Rather than backtracking to the previous node when a constraint conflict occurs, the algorithm back-jumps along the search path to the node that conflicts with the current node based on that constraint.[4] Conflict-directed back-jumping was proposed for Constraint Satisfaction Problems by Dr. Patrick Prosser in his groundbreaking 1993 paper.

(=> (== BatteryLevel LOW) (! = taskVariable curTaskID))
(Overridden by Charge Task)

For each path, if (path.traversable == false), *(! = PathVariable path)*

For each path, *(=> (== PathVariable path) (== PathLengthVariable path.length))*

For each path, *(=> (== PathVariable path) (== ObstructionVariable path.obstruction))*

Consistency is a property of a CSP that directly correlates to the effort required to solve a CSP, relating values and variable domains that may conflict with a constraint.

- **Node consistency (1-consistency): not any nodes that violate unary constraints upon those nodes.**
- **Arc consistency (2-consistency): all pairs of variables within the CSP there are not any values within either variable's domain that may cause a conflict.**
- **Path consistency (3-consistency): ensures that for any three variables there are not any possible assignments that may result in a conflict for any constraints between the variables**

Obtaining strong n-consistency is also NP-hard.[5] Constraint propagation techniques, like the ones described above, improve arc consistency—necessary to conform to strict regulations, which require strong consistency. The CSP solution can be evaluated according to the following five metrics, and solutions verified with a student's t-test:

- ***CSP solve time / CPU time***
- ***Task execution time***
- ***Resources consumed***
- ***Number of collisions***
- ***Number of failures***

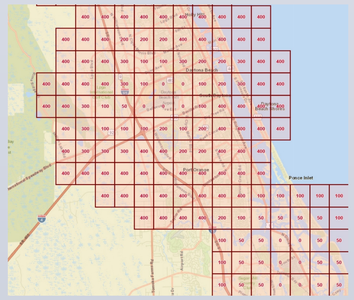


Figure 5: FAA UAS Arcgis

While rule-based systems currently dominate the market, applying constraint programming techniques by developing a decision maker can decrease computational times and resource expenditure up to an order of magnitude. The commercial applications of this research are wide and varied; having previously been applied in schedule planning and logistics, this research has direct applications in autonomous delivery. Further, these techniques can be used for any application in which computational time, space, or other resources are a concern; this has attracted further attention from emergency preparedness and response groups that use unmanned systems to respond in real-time to emergencies including crime scenes, train derailments, etc.

Unmanned platforms are increasingly being used for autonomous surveying and inspection of government infrastructure, an area in which personnel numbers and safety are concerns—concerns that the product of this research can mitigate. In the future, I plan to test further the platform with partners in industry, academia, and government, finish related publications, and continue researching constraint programming methods as I look forward to a research career.

References

[1] Akers E. L., Stansbury, R.S., et al, "Long-Term Survival of Mobile Robots" (2012)

[2] Dechter R., "Constraint Networks," in *Encyclopedia of Artificial Intelligence* (1991)

[3] Dechter R. & Pearl J., "Tree-clustering schemes for constraint-processing" (1988)

[4] Kumar, V., "Algorithms for Constraint Satisfaction Problems: A Survey" (1992)

[5] Mackworth, K., "Consistency in networks of relations" (1997)

[6] NIST, "Dictionary of Algorithms and Data Structures" (2017)

[7] Rossi, F., et al, "On the Equivalence of Constraint-Satisfaction Problems" (1989)

[8] Russel, S., & Norvig, P., *Artificial Intelligence: A Modern Approach* (2002)

[9] Stansbury, R.S. & Agah, A. Artif Intell Rev (2012) 38: 67.

Acknowledgements and Contact

BARRY GOLDWATER SCHOLARSHIP & EXCELLENCE IN EDUCATION FOUNDATION

courtneythurston.com
thurstc1@my.erau.edu
stansbur@erau.edu

Research Funded by:
Embry-Riddle Aeronautical University, Daytona Beach

PI Funded by:
Goldwater Scholarship, Jack Kent Cooke Foundation, Burger King Scholars Foundation, GE-Reagan Foundation, Coca-Cola Scholars Foundation, SanDisk Scholars, Elks Most Valuable Student Competition Scholarship, AXA Achievement Foundation, VIP Women in Tech Scholarship, Brad Feld and Amy Batchelor Aspirations in Computing Scholarship, National Space Club, Lint Center Army Staff Sgt. Richard S. Eaton, Jr. Scholarship, GoEnnounce & UPromise Define Yourself Scholarship, Harry E. Arcamuzi Aviation Scholarship, CCA Scholarship, Presidential Scholarship, Women of Excellence Scholarship, FIRST Robotics Scholarship, Alumni Endorsement Grant, Campus Travel Grant, NCWIT Aspirations in Computing Award

www.PosterPresentations.com