

Design and Implementation of Meta-Heuristics for Constrained Optimization Problems

Abstract

It is highly desirable in many industrial and business settings to maximize output or profit while respecting the constraints of the underlying production system, such as the number of hours a machine is allowed to be used per day, or the size of containers used to transport goods. Such problems get exponentially more difficult to solve as the problem size grows linearly, making it more and more difficult for humans to solve them by hand. The class of problems with this exponential growth property (as well as several other properties) have been labeled by computer scientists as NP-Hard. Because these problems are difficult to solve using simple algorithms, and because they have many practical applications, they are interesting from both practical and theoretical viewpoints.

For our project we chose two meta-heuristics: **simulated annealing**, and **genetic algorithms**. We chose two constrained optimization problems to apply these algorithms to: the **bin packing problem** and the **generalized assignment problem**. While the bin packing problem is easier to visualize and solve because it is combinatorially less complex, the generalized assignment problem is more interesting because it is combinatorially more complex, and (not incidentally) has more practical applications.

Our project placed additional focus on two important parts of developing algorithms for constrained optimization problems: **visualization** and **simulation**. We developed a graphical user interface that allows the user to see algorithms solve problem instances in realtime, and allows the user to schedule simulations on a remote computer. We used these tools to run simulations comparing various simulated annealing and genetic algorithms, and compared their performances to previous work, which we hope provides some insight into what makes these algorithms successful.

Group 11

Authors:

Rob Gallan (ESE)

Raymond Hsu (CSE)

Advisors:

Dr. Steven Kimbrough (OPIM)

Dr. Lyle Ungar (CIS)

Demo Times:

11:00-12:00

1:30-3:00

Bin Packing Problem Simulation Setup:
The user chooses what type of problem she would like to run, specifying things such as the **capacity** and **number of items**.

Genetic Algorithm(GA) Setup:
The user chooses the parameters of the GA, such as the **mutation rate**, a parameter which determines how different new solutions are from old solutions.

Simulated Annealing Setup:
The user chooses what the cooling schedule should look like. The cooling schedule regulates how much time the algorithm spends exploring new areas of the solution space versus exploiting known good solutions.

Visualization

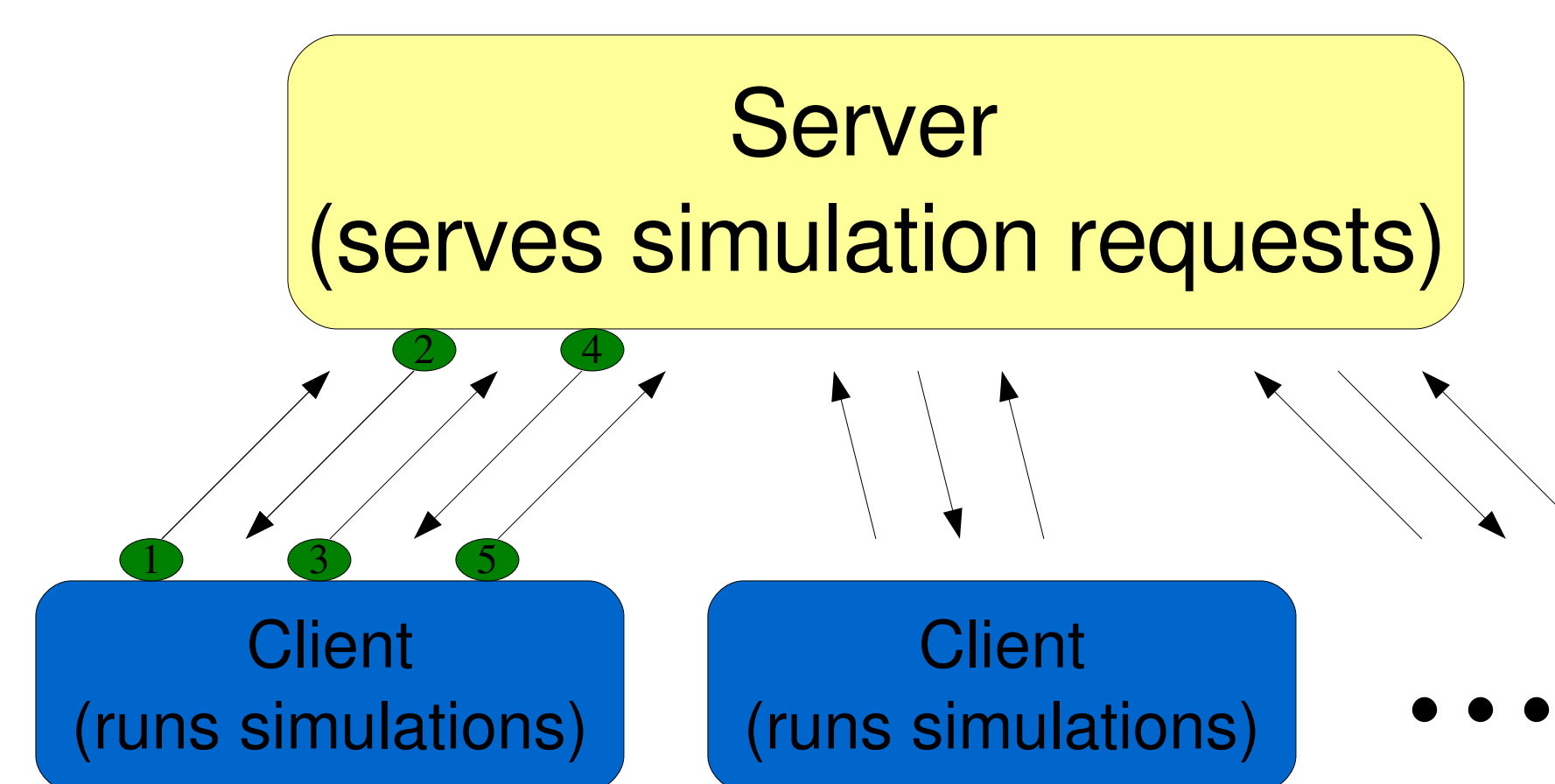


Fitness Function Plot:
This panel gives an estimate of how good each of the algorithms is doing at any point in time. It does this using a measure called a **fitness function**, which is a proxy for the **objective function** (in this case the **number of bins**). The purpose of the fitness function is to provide the algorithm with more information than the objective function about how good a solution is.

Solution Plot:
This section plots the current best solution for each of the algorithms. Each colored block represents an item, and each vertical column represents a grouping of items, called a bin. The goal is to group the items into the the fewest number of bins (fewer bins corresponds to fewer horizontal rows here).

Simulation

Setup



- Client asks if it has most recent simulation code
- Server sends updated simulation code if necessary
- Client asks for a simulation to run
- Server sends the configuration information for the next required simulation
- Client sends back the results of the simulation and the process repeats

Setup Summary:
Many simulations must be run in order to compare the performance of these algorithms across different configurations, problem instances, and initial conditions. In addition they must be run for multiple trials to test repeatability. In total we ran more than 350 different simulations, which would have taken almost a month running continuously on one computer. In order to complete all these simulations in a reasonable amount of time, multiple computers were used. By automating the scheduling of the simulations, additional time can be saved, and the data can be gathered and analyzed more quickly.

The Configurations:
Each row of the table represents one unique type of genetic algorithm. Each character in the labels in the Configuration column corresponds to one trait of the algorithm. For example, the first character of each configuration is either an 'M'(maximize profit) or an 'R'(random), depending on whether this configuration uses a profit maximization technique to create the initial population of solutions, or whether it starts with random solutions. It is clear from this table that profit maximization at initialization is advantageous, as the algorithms that perform better generally use maximization.

This table not only shows which configurations perform best, but more importantly, it shows which traits are most advantageous, which helps direct further research. For example, in these simulations it made little difference whether the second trait, the parent selection method, was either Binary Tournament Selection, or Two(2) Population Selection. This could indicate that how the parents of new solutions are selected is not as important as other traits in determining the overall success of the algorithm.

Sample Results

The Generalized Assignment Problem:

This table summarizes the performance of several types of genetic algorithms in solving the generalized assignment problem. The entries in the table represent the quantity (best solution found/ known best solution) averaged over 5 trials, so if the algorithm found the best known solution, the entry would be 1.000. The problem difficulty increases from left to right. It is interesting to note that several of the configurations perform better (relatively) on easier problems, but their performance decreases on more difficult problems (and vice versa). This data was collected using a set of 16 desktop computers over an 8 hour period.

Configuration	Average	Agents															
		5	10	15	20	24	32	48	60	100	100	100	100	100	100	100	
M2WSI	0.97263	1.000	0.999	0.999	0.999	0.999	0.994	0.996	0.998	0.989	0.958	0.941	0.935	0.949	0.887		
MBWS-	0.97130	1.000	0.997	0.999	0.999	0.993	0.996	0.996	0.984	0.967	0.943	0.935	0.939	0.880			
MBWSI	0.97111	1.000	0.998	0.999	0.999	0.995	0.995	0.995	0.980	0.964	0.943	0.939	0.940	0.876			
MBW2I	0.97039	1.000	0.998	0.997	0.999	0.995	0.995	0.995	0.987	0.962	0.948	0.934	0.945	0.861			
M2SS-	0.97045	1.000	0.996	0.999	0.999	0.993	0.996	0.997	0.986	0.954	0.948	0.937	0.942	0.869			
MBSSI	0.97048	1.000	1.000	0.999	0.998	0.993	0.995	0.996	0.992	0.957	0.957	0.935	0.942	0.872			
M2SS-	0.97008	1.000	0.999	0.997	0.998	0.994	0.995	0.996	0.989	0.958	0.932	0.935	0.944	0.879			
M2WS-	0.97009	1.000	0.998	0.999	0.999	0.994	0.995	0.997	0.986	0.952	0.935	0.933	0.943	0.879			
M2W2I	0.97007	1.000	0.999	1.000	0.999	0.994	0.995	0.997	0.989	0.959	0.948	0.933	0.940	0.863			
MBW2-	0.96942	1.000	0.999	0.998	0.998	0.993	0.994	0.996	0.987	0.959	0.939	0.936	0.939	0.864			
MBSS-	0.96932	1.000	0.998	0.997	0.999	0.992	0.996	0.996	0.986	0.949	0.932	0.938	0.943	0.876			
M2S2I	0.96889	1.000	0.995	0.999	0.997	0.993	0.994	0.996	0.989	0.955	0.937	0.933	0.941	0.867			
RBWS-	0.96929	1.000	0.998	0.999	0.999	0.994	0.995	0.995	0.995	0.960	0.934	0.932	0.942	0.861			
RBWSI	0.96937	1.000	0.999	0.999	0.999	0.994	0.995	0.996	0.972	0.957	0.934	0.931	0.945	0.881			
RBSSI	0.96894	1.000	0.997	0.999	0.998	0.994	0.995	0.997	0.977	0.957	0.942	0.933	0.943	0.866			
R2WS-	0.96887	1.000	1.000	0.999	0.998	0.995	0.996	0.996	0.969	0.961	0.940	0.934	0.946	0.862			
M2W2-	0.96860	1.000	0.998	0.998	0.999	0.995	0.995	0.996	0.986	0.951	0.935	0.933	0.942	0.865			
M2SS-	0.96822	1.000	0.999	0.997	0.999	0.994	0.995	0.997	0.985	0.957	0.924	0.934	0.933	0.865			
R2W2-	0.96810	1.000	0.999	0.996	0.999	0.997	0.995	0.996	0.978	0.961	0.930	0.931	0.943	0.863			
RBW2I	0.96783	1.000	0.997	0.999	0.998	0.995	0.995	0.997	0.974	0.956	0.932	0.933	0.943	0.863			
RBSS-	0.96766	1.000	0.997	0.999	0.997	0.994	0.996	0.996	0.973	0.946	0.934	0.934	0.947	0.867			
MBS2I	0.96753	1.000	0.999	0.998	0.997	0.994	0.995	0.995	0.972	0.963	0.925	0.927	0.943	0.860			
R2WSI	0.96781	1.000	0.998	0.999	0.999	0.994	0.995	0.997	0.968	0.946	0.936	0.934	0.939	0.876			
R2SSI	0.96747	1.000	0.998	0.997	0.999	0.993	0.993	0.996	0.969	0.941	0.937	0.935	0.943	0.873			
R2W2I	0.96776	1.000	1.000	0.998	0.999	0.994	0.996	0.997	0.979	0.945	0.935	0.932	0.941	0.865			
R2S2-	0.96707	1.000	0.998	0.999	0.999	0.993	0.996	0.996	0.971	0.941	0.925	0.932	0.944	0.878			
RBW2-	0.96698	1.000	0.996	0.998	0.999	0.996	0.996	0.997	0.966	0.943	0.934	0.934	0.942	0.871			
R2S2I	0.96654	1.000	0.998	0.998	0.999	0.993	0.995	0.995	0.970	0.943	0.930	0.931	0.944	0.868			
R2SS-	0.96590	1.000	0.999	0.998	0.998	0.994	0.996	0.997	0.967	0.949	0.939	0.935	0.942	0.874			
MBS2-	0.96525	1.000	0.999	0.996	0.998	0.994	0.995	0.996	0.981	0.952	0.935	0.930	0.935	0.859			
RBSS-	0.96264	1.000	0.998	0.999	0.999	0.993	0.995	0.995	0.963	0.947	0.938	0.938	0.939	0.851			
RBSSI	0.96155	1.000	0.997	0.996	0.999	0.992	0.995	0.996	0.952	0.940	0.933	0.935	0.935	0.846			
MAX		1.000	1.000	1.000	0.999	0.997	0.997	0.998	0.992	0.967	0.948	0.938	0.949	0.887			
MEDIAN		1.000	0.998	0.999	0.999	0.994	0.995	0.996	0.980	0.956	0.934	0.933	0.942	0.868			
MIN		1.000	0.995	0.996	0.997	0.992	0.994	0.994	0.980	0.939	0.918	0.920	0.936	0.846			
STDDV		0.00000	0.00118	0.00094	0.00067	0.00106	0.00058	0.00076	0.00938	0.00793	0.00843	0.00367	0.00287	0.00912			