Intelligent Information Management, 2017, 9, 97-113
http://www.scirp.org/journal/iim
ISSN Online: 2160-5920
ISSN Print: 2160-5912

Decision Support through Intelligent Agent Based Simulation and Multiple Goal Based Evolutionary Optimization

Wissam Alobaidi[1*], Eric Sandgren[1*], Entidhar Alkuam[2]

[1]Systems Engineering Department, Donaghey College of Engineering & Information Technology, University of Arkansas at Little Rock, Little Rock, USA
[2]Department of Physics and Astronomy, College of Arts, Letters, and Sciences, University of Arkansas at Little Rock, Little Rock, USA
Email: *wmalobaidi@ualr.edu, *exsandgren@ualr.edu

How to cite this paper: Alobaidi, W., Sandgren, E. and Alkuam, E. (2017) Decision Support through Intelligent Agent Based Simulation and Multiple Goal Based Evolutionary Optimization. *Intelligent Information Management*, **9**, 97-113.
https://doi.org/10.4236/iim.2017.93005

Received: March 17, 2017
Accepted: May 23, 2017
Published: May 26, 2017

Abstract

Agent based simulation has successfully been applied to model complex organizational behavior and to improve or optimize aspects of organizational performance. Agents, with intelligence supported through the application of a genetic algorithm are proposed as a means of optimizing the performance of the system being modeled. Local decisions made by agents and other system variables are placed in the genetic encoding. This allows local agents to positively impact high level system performance. A simple, but non trivial, peg game is utilized to introduce the concept. A multiple objective bin packing problem is then solved to demonstrate the potential of the approach in meeting a number of high level goals. The methodology allows not only for a systems level optimization, but also provides data which can be analyzed to determine what constitutes effective agent behavior.

Keywords

Decision Support, Multiple Goal, Agent Based, Genetic Optimization, Bin Packing

1. Introduction

Over the past ten years, there has been an increasing interest in agent based modeling and simulation. Summaries of application to manufacturing, planning and scheduling are given by Shen *et al.* [1], Marik [2] and Yoo [3]. An agent refers to a software segment that performs a specific task. This task may be simple or complex and the agent may be intelligent or not. An intelligent agent refers to

one that has some form of reasoning and learning capability. The agent taxonomy allows for a model to be developed for systems that involve complex behavior. When the model is integrated with a global optimization algorithm, the ability to improve performance at both a local and systems level becomes possible. Often, the decisions which are made by a local agent are based on experiential data which may have a very local context. System complexity often makes it extremely difficult to look forward in time or to consider global rather than local objectives. This leads to inefficiencies, which if avoided, could improve the operational efficiency of the system. In industry segments involving logistics and manufacturing, this efficiency gain may make the difference between operating at a profit or loss. As the fixed cost of operation increases (energy, labor and raw materials), the only way to remain profitable is through increasing operational efficiency. An evolutionary optimization algorithm may be utilized to assist the local decision making capability of the agents to achieve this system goal. This combination represents an innovative idea and holds the promise of a significant gain in productivity over a wide range of industries.

The strengths and weaknesses of agent based modeling coupled to mathematical optimization for a situation involving multiple objective distributed resource allocation in a dynamic environment are reviewed by Davidsson *et al.* [4]. Narzisi *et al.* [5] optimized an agent based model for emergency response planning where the optimization was performed by an evolutionary algorithm. Optimization applied to parameter tuning of the agents has been performed with both a single objective by Calvez [6] and multiple objectives by Rogers [7]. Specific instances of the utilization of agent based simulation in conjunction with various optimization approaches have been presented by Deshpande [8] and Gjerdrum *et al.* [9] for manufacturing scheduling, Sirikipanichkul [10] for freight hub location, Neagu [11] for transport logistics and by Botterud [12] for expansion in electricity markets. It is clear from the wealth of research activity in the application of agent based modeling and simulation that the approach offers a valid means for addressing complex, real world problems. The difficulty in most applications involves how to best utilize optimization as a means of improving the performance of the system being modeled. Intelligence at the local and global level is difficult to implement through experience or rule based approaches. The value of coupling of a genetic optimizer directly to the individual agents is investigated herein.

Agents are often introduced as elements which model the behavior of humans performing a specific task. Many such tasks represent highly skilled positions such as a scheduler for a line of a manufacturing plant or a loader for a freight hub. For operations involving complex reasoning, it is difficult to develop consistent performance under varying day to day conditions. This is particularly true when such reasoning must include local and global objectives. The ability to share the intelligence between an agent's knowledge base and an optimization algorithm helps to insure that all goals are factored into the solution and the entire solution space is investigated. The results from such an optimization in

practice can help improve the behavior of a local agent over time and some of the intelligence supplied through the optimization can be incorporated into the logic employed by the agent. This allows a framework to be built for a specific application class for which the performance can be continuously improved over time.

The particular implementation considered in this study will involve simple agents performing tasks that involve a certain level of reasoning. The agent in each system is coupled to an evolutionary or genetic optimization algorithm. The optimization algorithm utilizes a goal programming formulation in order to address multiple objectives. Two examples are presented which allow for the approach to be developed and demonstrated. The first example involves the solution of a simple, but non-trivial, peg game where the agent must make all move decisions with a global goal of solving the game. The second example involves a multiple objective bin packing problem with the agent being responsible for packing a rectangular trailer with a prescribed set of shipments or loads. Both examples involve only a single agent, but the approach is scalable to include any number of agents. The approach leads to a distributed rule based decision support environment that can be applied to complex problems encountered over a wide range of industries. The contribution of this work is in the demonstration of how a combination of evolutionary optimization methods and agent based simulation can solve decision support problems.

2. An Evolutionary Goal Programming Approach

For a decision support system implementation utilizing intelligent agents, the agents must be capable of making a sequence of decisions at the local level which determine not only local but global system behavior and performance. The impact of each individual decision is influenced by all previous decisions and all future decisions are impacted as well. The traditional nonlinear mathematical programming formulation for this problem class is defined as:

$$\text{Maximize } f(x); x = \left[x_1, x_2, x_3, \cdots, x_n\right]^{\mathrm{T}} \tag{1}$$

Subject to

$$g_j(x) \geq 0;\ j = 1, 2, \cdots, J \tag{2}$$

$$h_k(x) = 0; k = 1, 2, \cdots, K \tag{3}$$

and

$$x_i^{low} < x_i < x_i^{high} \tag{4}$$

In this formulation, $f(x)$ represents a measure of global performance which is to be maximized. This measure may be linear or nonlinear and related to efficiency, profit or any calculable quantity. The design or decision vector, x, is composed of all decision possibilities that each agent must make at the local level. Additional design variables may be included to allow for evaluation of non-agent based decision related factors. The design variables may be continuous, integer or discrete and may be defined on a range as by Equation (4), or as a set of dis-

crete or integer valued choices. The number of assigned decision values may vary for each individual decision variable and the design variable vector includes the local decisions for all local agents. Equations (2) and (3) define constraints which establish the feasible search space. The greater or less than constraint given by Equation (2), allows for a constraint which can be over satisfied, while the equality constraint given by Equation (3) specifies a constraint which must be exactly satisfied.

The mathematical formulation presented above is useful for the solution of a wide range of problems. Difficulties arise, however, when multiple objectives are desired as well as soft constraints. Soft constraint represent conditions which are desirable but are allowed to be violated if beneficial trade-offs may be made among the values of the other objectives. A goal formulation may be applied to the decision support problem to easily include these additional aspects. A goal programming formulation for the broad class of decision support problems may be expressed as follows:

$$\text{Maximize } f(x) = \sum_{i=1}^{I} W_{kj} P_k \left\{ d_i^+ + d_i^- \right\} \tag{5}$$

for $k = 1, 2, \cdots, K$

$$j = 1, 2, \cdots, J$$

where the d_i^+ and d_i^- are positive valued variables and represent the under and over achievement of goals which may be specified by one of the following goal constraint equation forms:

$$G_j(x) + d_i^- = b_i \tag{6}$$

$$G_j(x) - d_i^+ = b_i \tag{7}$$

and

$$G_j(x) - d_i^- + d_i^+ = b_i \tag{8}$$

From a comparison to a linear programming formulation, the d_i^- and d_i^+ deviational variables may be thought of as a form of slack variables. With this analogy, the first two goal constraints may be seen to be less than or equal to and greater than or equal to forms. The third form essentially represents a hard equality constraint, which allows for absolute requirements to be met. In a goal formulation, only the deviational variables appear in the objective function as represented by Equation (5). Each deviational variable, however, is a function of the decision variables. The specific form of the objective function allows for weighting factors, W_{kj}, and for goal priorities P_k for each goal. There is no limit to the total number of goals allowed. An evolutionary optimization algorithm may be implemented to address either the formulation given by Equations (1)-(4) or for that given by Equations (5)-(8). The general flow of the algorithm is presented in Figure 1.

3. The Peg Game: A Simple Example

The implementation of an optimization controlled, agent based decision support

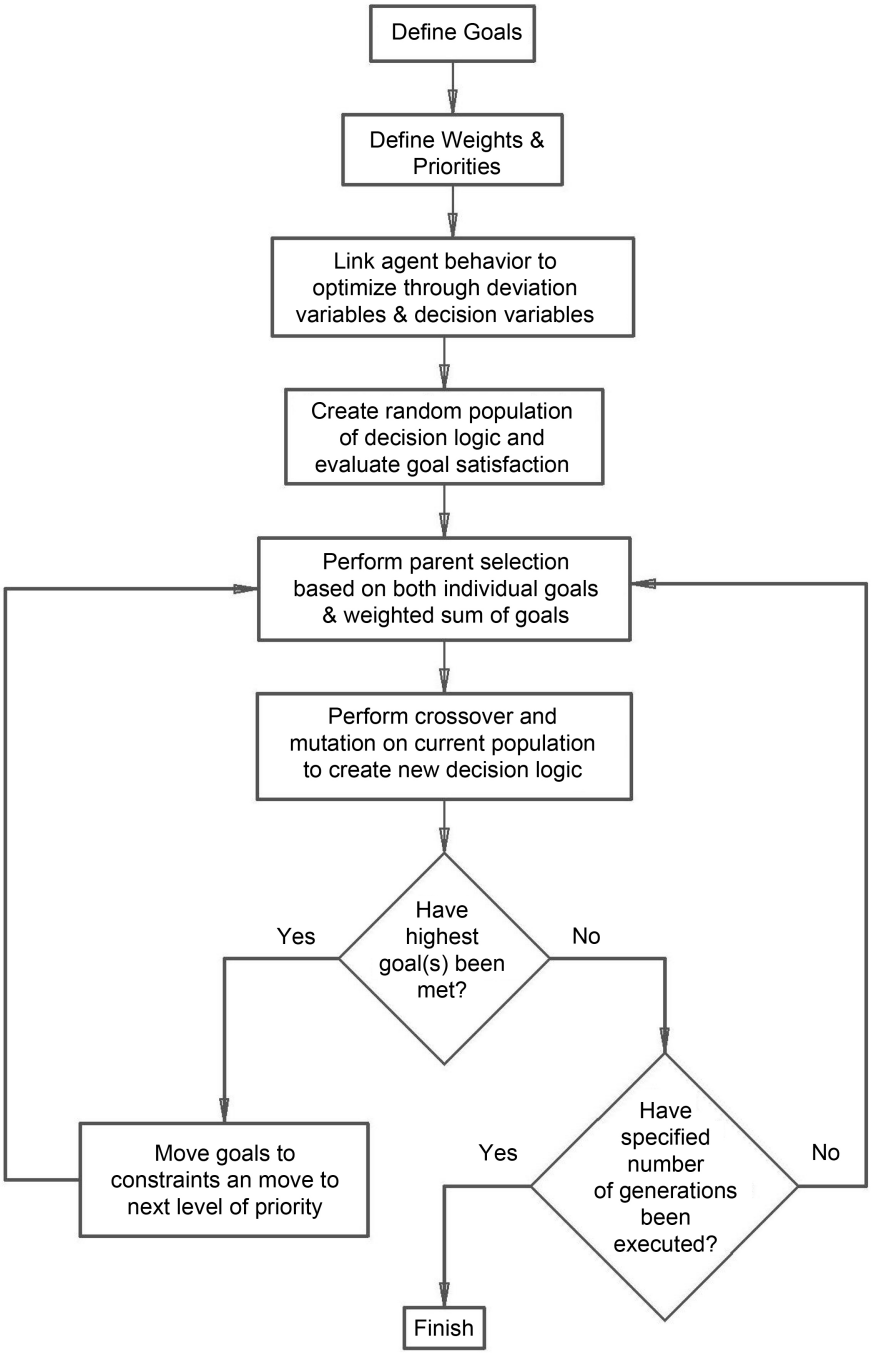

Figure 1. Flowchart of evolutionary goal programming approach.

application is best demonstrated through the consideration of a simple, but non-trivial, example. The example utilized here involves a peg game which has a simple measure of success, but no clear-cut strategy to implement in a given game situation. The game consists of fifteen uniformly spaced holes on a game board that is configured as an equilateral triangle. The game layout is presented in Figure 1. The holes on the game board are filled with pegs which can be removed according to the rules of the game. The game begins as the user removes one peg from the board and then continues by jumping pegs as allowed by the

current configuration. A peg may be jumped only if a peg is directly adjacent to it and the opposite position is vacant. Once a peg has been jumped, that peg is removed from the game board and the process continues. The game terminates when all possible moves have been made. If a single peg remains, the game is considered to be won. As with most decision support problems, the current game condition at any point in the solution process is determined by all previous moves. At each step in the game, however, an appropriate move may be determined by an agent whose intelligence is derived from an evolutionary optimization algorithm.

The agent, which is implemented in the game simulation, must make the decisions which control the operation of the game. At each game state, the agent is responsible for selecting which peg to move among all possible pegs that can move. This is a convenient, low level implementation of agent based modeling where the agent only has the tasks of identifying possible moves and selecting one at each game state. The mathematical description of the optimization problem would be difficult without the agent based simulation. This is because the determination of possible moves, the selection of a move at each game state and even the number of moves required to complete an individual game simulation vary from simulation to simulation. The mathematical formulation for this particular implementation may be expressed as follows:

$$\text{Minimize } f(x) = \text{Number of pegs remaining at stage } p \tag{9}$$

where

$$x = \left[x_1, x_2, x_3, \cdots, x_{14}\right]^{\mathrm{T}} \tag{10}$$

And

$$x_1 = \text{first peg removed to start the game; } x_1 \in \{1, 2, 3, \cdots, 15\} \tag{11}$$

$$x_n = \text{peg movement option from list of possible moves at game simulation step } n,\ n = 1, 2, 3, \cdots, 14;\ x_n \in \{1, 2, \cdots, n_{\text{moves}}\} \tag{12}$$

and n_{moves} is defined as the maximum moves available at any game state and p is defined as the game state where no more moves are available. Note the design variables in the encoding are integer in nature and are only known as the simulation progresses from game state to game state. Since the only goal is to minimize the number of pegs remaining after all moves have been made, the formulation given by Equations (9)-(12) is equivalent to the formulation given by Equations (1)-(4). The agent utilization, coupled with an evolutionary optimization algorithm allows for a straightforward implementation of the intelligent decision support structure.

With most agent implementations, intelligence can be introduced to guide the decision making of the agent. The difficulty with this approach is that the impact of local decisions made at each game state are difficult to relate to the final game execution result. This situation is common in a wide range of applications where local decisions, made with local information may lead the enterprise far from the global or system optima. Specific examples include trailer packing for a LTL op-

eration, truck fleet routing, pick-up and deliveries, home care medical staff scheduling and routing, and many others. The efficacy of local decision making can be improved through the coupling to an evolutionary optimization algorithm which is capable of making local decisions in a global, multiple goal objective context. A post optimal analysis can be executed to extract key characteristics of sound local decisions that led to the optimal global solution. This information can be utilized to improve the local intelligence of the agent. Over time the decisions at the local level can be split between the agent acting independently and the genetic algorithm enforcing decisions.

An evolutionary or genetic algorithm is well suited to deal with the general agent based decision support problem. It can easily handle integer valued variables as well as a variable number of moves, or game steps in this case, for each simulation or objective function evaluation. Another positive feature of the algorithm is that it can handle solution spaces with many local minima which is common in decision support applications. Coupled with a goal programming formulation, multiple objectives can be handled which is another common trait of this problem class. The optimization begins with randomly selected decisions for the agent at each move and continually refines the decisions until a single peg remains at the end of a game simulation. These decisions constitute the optimal decisions for the game playing agent. The decisions can then be evaluated for global characteristics that could possibly be built into the stand alone agent intelligence in future implementations.

The procedure described above was implemented for the fifteen peg game and a solution was generated within thirty generations with a population of two hundred decision strategies for the game agent in each generation. In order to track the solution, the holes on the game board are numbered and this numbering scheme is shown in Figures 2-4 document the final decision strategy utilized by the agent in a move by move fashion.

The post optimal analysis for this example is difficult due to the elementary nature of the decision options. However, it can be noted that the optimal decision strategy did seem to be centered around both keeping the number of choices at each step as large as possible and to maintain some semblance of grouping

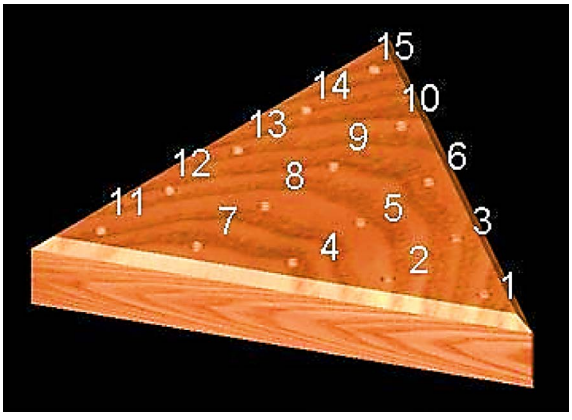

Figure 2. The peg game board.

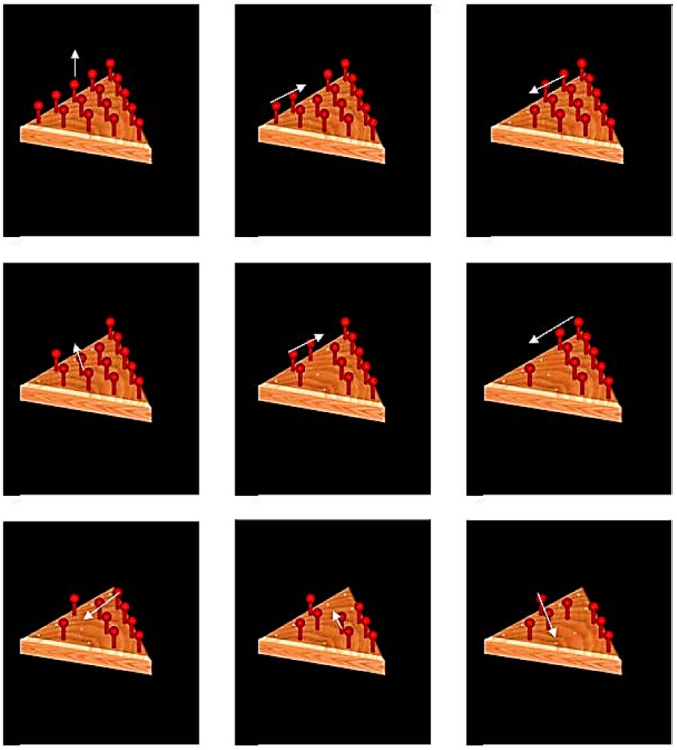

Figure 3. First 9 moves for the 15 peg game solution.

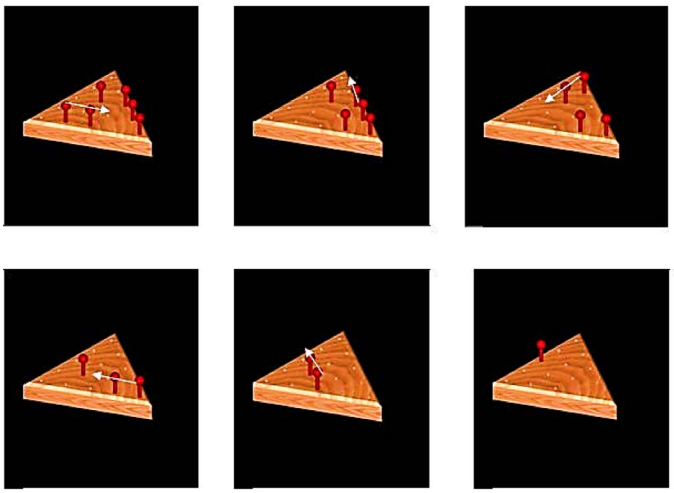

Figure 4. Final 6 moves for the 15 peg game solution.

of the pegs at each game step. Both of these characteristics are aligned with the anticipated solution strategy for the peg game. As the applications become more complex, this simple correlation between decision strategy and results is not expected, but useful information may be gathered as a number of scenarios are executed.

4. Multi-Objective Bin Packing: A More Challenging Example

The bin packing problem fits in well with the decision support framework where agents are utilized with partial intelligence supplied by an evolutionary optimi-

zation algorithm. Both the two and three dimensional bin packing have been studied extensively, and remains as NP hard problems. Solution approaches have been generated through simulated annealing by Rao [13] as well as through the application of genetic algorithms [14] [15] [16] [17]. None of these solution approaches considered a multi-objective solution. An agent based approach was proposed by Lau [18], and heuristic approaches by Epstein [19] and Lim [20] provide insight to potential agent based logic and behavior. In a multi-objective form, the problem formulation can be utilized to address critically important aspects of the transportation industry. The operation of a less than full load (LTL) business involves the packing of semi-trailers with individual shipments which are then transported through a series of hubs. At each hub, shipments are generally unloaded from one trailer and reloaded in another. Upon reaching the final hub destination, the shipment is delivered locally through a separate operation termed pick-up and delivery. One of the highest skill labor jobs is the loader as the efficiency of the business is highly dependent on the placement of each shipment in the trailer as well as the load fill of the trailer before leaving the hub. The loader in this example will be represented by an agent. The genetic optimization algorithm will provide global decision making ability to the loading agent. The traditional bin packing solution would focus on trailer fill, but in this example other factors or goals are considered as well. Among these factors are placement of top-loads, priority loads, front loading common hub destination shipments and the center of gravity of the trailer after loading.

The maximum fill condition is astraightforward one. Trailer fill refers to the amount of total volume in the trailer filled by shipments. Any unfilled space represents lost efficiency, as the trailer will execute the appointed route whether or not it is completely filled. As trailer fill increases, it potentially reduces the number of trailers required to transfer all shipments through the hub network to their final destinations. The only caveat is that the trailer may weigh out, rather than cube out. The term cube out refers to filling the trailer volume to capacity, while the term weigh out refers to the total loaded weight of the shipments being equal to the maximum allowed, even though there may be unused volume. LTL shipments may be pallet based or individual items, and as a general rule, most are rectangular in shape. Top load shipments are those that are fragile and cannot have any other shipments placed on top of them. These loads will generally be placed at the very top of the trailer. Priority loads are those shipments which must progress through the hub network as quickly as possible. An additional charge is associated with priority shipments and this charge is only justified if the shipment arrives when promised. In order to facilitate a reduction in time for priority loads, they are ideally placed near the rear of the trailer so they may be unloaded and reloaded as soon as a hub is reached.

Front loading is a term which is used when a number of shipments with the same hub destination following the current one, or the same final destination are loaded in the front of the trailer. This allows the trailer to be unloaded, leaving these shipments untouched as the inbound trailer becomes the outbound trailer

to the next destination. Each reduction in handling for a shipment reduces the risk of damage as well as a savings in time and labor. Finally, the center of gravity of a loaded trailer is important. If the center of gravity is too high or not centered in width or toward the front in length, a bad handling condition may occur. The possibility of a roll over as a corner is traversed at speed is also a possibility. Each condition added beyond the traditional maximum fill adds complexity to the optimization problem solution. On the other hand, significant value is added to the transportation firm. A single trailer example is considered here in order to demonstrate the approach. The extension to all trailers in service at an LTL is certainly possible, particularly with the parallel nature of the process.

In order to model the agent based trailer loading example, the trailer is subdivided into a series of small rectangular solids. The size of each rectangular solid may be decreased in order to increase the accuracy along with the associated level of computational complexity for solving the problem. In this example a grid of 4 × 5 rectangular solids is introduced at each of 12 individual sections along the length of the trailer as shown in Figure 5. This subdivision results in a total of 240 individual loading volumes. With this simplification, a set of loads or shipments may be placed in the trailer by the loading logic resident in the local loading agent. To take the example a step further, consider the set of shipments listed in Table 1. Each of the individual 23 shipments has dimensions, weight, next hub destination, priority and an indication of the shipment being a top load only load. The next hub destination (7 total) represents the hub the shipment is to be sent to after processing at the current hub destination for the trailer. The priority (1, 2 or 3), is an indication of a shipment which must be processed as quickly as possible as it is a premium shipment. The shipments with a priority of one are the highest priority shipments. In this example, the total volume of all shipments is equal to the total trailer volume.

The goals utilized in the formulation for this example are as follows: The total volume fill should be as great as possible. All top loads should have no load placed on top of them. The high priority loads should be located in the rear of the trailer and shipments with the same next destination should be consolidated as a head load in the front of the trailer. Finally, the center of gravity should be as low as possible, as near to the center of the trailer width as possible and as far forward in the trailer as possible. The goal programming formulation for this

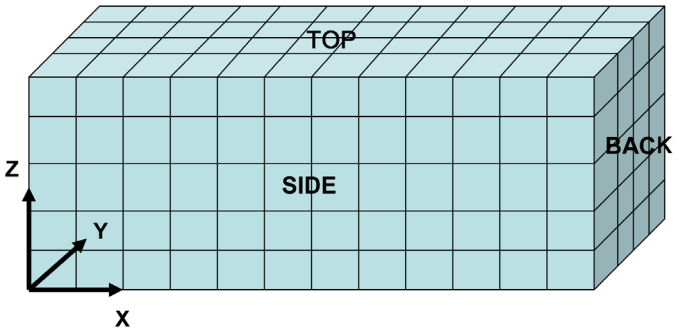

Figure 5. Trailer volume division.

Table 1. Shipment data for example problem.

Load	Hub Destination	Weight	Priority	Top Load	Dimensions (x/y/z)
1	2	10.0	2	No	4/3/2
2	3	8.5	3	No	4/5/2
3	3	7.5	2	No	4/2/2
4	4	5.0	1	No	2/2/2
5	2	5.0	1	No	2/2/2
6	2	8.0	2	No	2/2/2
7	3	5.0	2	No	2/2/2
8	4	2.0	2	Yes	1/1/1
9	2	2.0	3	Yes	1/1/1
10	5	2.0	3	No	1/1/2
11	6	2.0	3	No	1/1/2
12	7	2.0	2	Yes	1/1/2
13	2	3.0	2	No	2/2/4
14	5	3.0	3	No	2/2/3
15	8	3.0	2	No	2/3/1
16	9	5.0	1	Yes	2/1/2
17	7	2.0	2	Yes	2/1/1
18	3	4.0	2	No	2/3/4
19	5	2.0	2	Yes	2/2/2
20	6	2.0	2	Yes	2/2/2
21	3	9.0	2	No	2/5/2
22	4	3.0	2	No	2/4/2
23	8	3.0	2	Yes	2/1/2

problem statement is as follows:

$$\text{Maximize } f(x) = W_1 d_1^+ + W_2 d_2^+ + W_3 d_3^+ + W_4 d_4^+ + W_5 d_5^+ + W_6 d_6^+ + W_7 \left(d_7^+ + d_8^-\right) + W_8 d_9^+ \tag{13}$$

$$G_1(x) + d_1^- = 0.95; \text{ where } G_1(x) = \frac{\text{Trailer Fill}}{\text{Total Trailer Volume}} \tag{14}$$

$$G_2(x) - d_2^+ = 0; \text{ where } G_2(x) = \text{Number of Covered Top Loads} \tag{15}$$

$$G_3(x) - d_3^+ = 0; \text{ where } G_3(x) = \text{Inaccessible Priority 1 Shipments} \tag{16}$$

$$G_4(x) - d_4^+ = 0; \text{ where } G_4(x) = \text{Priority Shipments in Front Half Of Trailer} \tag{17}$$

$$G_5(x) + d_5^- = 0.25; \text{ where } G_5(x) = \frac{\text{Front Load Fill}}{\text{Trailer Volume}} \tag{18}$$

$$G_6(x) - d_6^+ = 0.33; \text{ where } G_6(x) = \frac{CG_z \text{ Location}}{\text{Trailer Height}} \tag{19}$$

$$G_7(x) - d_7^+ + d_8^- = 0.50; \text{ where } G_7(x) = \frac{CG_y \text{ Location}}{\text{Trailer Width}} \tag{20}$$

$$G_8(x) - d_9^+ = 0.50; \text{ where } G_8(x) = \frac{CG_z \text{ Location}}{\text{Trailer Length}} \tag{21}$$

The above formulation may look complex, but it is actually a straightforward statement of the goals of the trailer loading problem. The first goal constraint states that the loaded percentage of the trailer should be at least 95 percent. Any shortfall is represented by the underachievement deviational variable d_1^+ which appears in the objective function to be minimized. No penalty is associated with a fill greater than 95 percent as expected. The underachievement variable appears in the objective function with a weight W_1 which allows it to have a high priority (W_1 large or a smaller priority) (W_1 smaller or equal to other goal weights). The second goal constraint states that no top loads may be covered or the underachievement variable d_2^+ will negatively impact the objective function by the number of covered top-loads multiplied by weighting factor W_2. Using a similar interpretation, the third goal constraint states that all priority one loads must be immediately accessible on opening the trailer. The fourth goal constraint requires priority level two shipments to be located in the real half of the trailer. The fifth goal constraint requires a front load fill with a common next hub destination to be at least 25 percent of the total volume. The sixth, seventh and eighth goal constraints require the CG location to be lower than or equal to 1/3 the trailer height, as close to the 1/2 the width (trailer center) as possible and closer to the front of the trailer than the rear. Any deviation from the desired goal constraint specifications results in a contribution to the objective function which penalizes that particular loading placement. It is extremely likely that no loading placement will satisfy all goals. However, the algorithm will do what it can to minimize underachievement of critical goals. The relative magnitude of the weighting factors may be adjusted to see what trade-offs are available among the various goal satisfaction levels. A priority based formulation may also be implemented which requires higher priority to be completely satisfied before lower priority goals may be considered.

The formulation represented by Equations (13)-(21) was coded and a solution was attempted with an evolutionary optimization algorithm. In order to execute the packing through an encoding which assists the loading agent implemented, the encoding structure was defined as follows:

$$x = \left[x_{1a}, x_{1b}, x_{1c}, x_{2a}, x_{2b}, x_{2c}, \cdots, x_{23a}, x_{23b}, x_{23c} \right] \tag{22}$$

The values which determine the placement of each individual shipment are ordered in triple valued groups. The first value of the three groups determines the loading order. The second group determines whether the shipment is placed starting from the right or left side of the trailer. The third determines the rotation (90 degrees) of the shipment before loading. The three tiered value structure is repeated for each shipment to be loaded in the trailer. The agent transforms the encoding values into a sequence of operations, as defined by the encoding, which determines the location of each shipment and the finished loaded configuration of the trailer. In order to place some of the intelligence at the agent, top loads were not allowed to have a load placed on top after they are placed. Also the agent logic prevents any two shipments from occupying the same volume

which is a difficulty with some bin packing algorithms.

The solution located by the intelligent agent based optimization is shown in Figures 6-10. Each figure gives a top, side and rear view of the loaded trailer as specified by the optimal solution located through the application of the goal programming formulation. The position of all loads may not be able to be discerned in all figures, but the key shipment placements are visible. Figure 6 simply presents the placement of each of the 23 individual shipments. The point of note here is that the trailer fill was 100 percent which actually overachieved the first goal constraint level. Figure 7 documents the fact that all top load shipments are located at the top of the trailer and have no loads placed on top of them. Once again, this was guaranteed by the intelligent agent placing the shipments. Figure 8 documents the fact that all priority one loads are immediately accessible once the trailer doors are opened and that while all medium priority

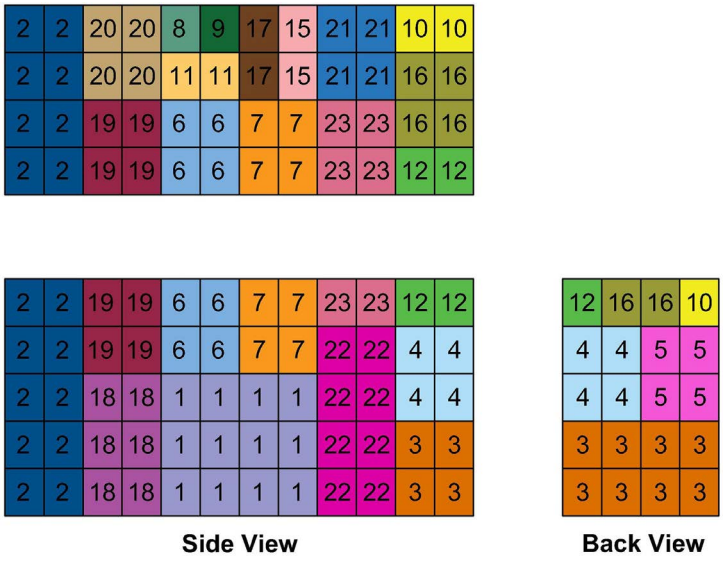

Figure 6. Shipment placement for optimal solution.

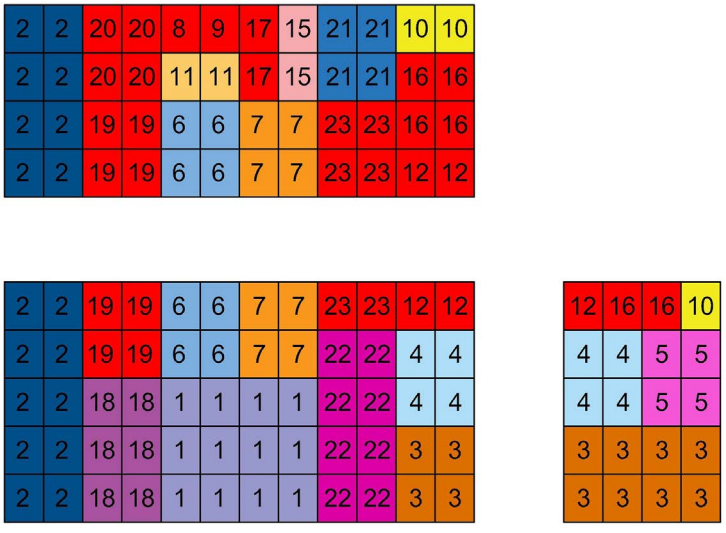

Figure 7. Top load placement for optimal solution.

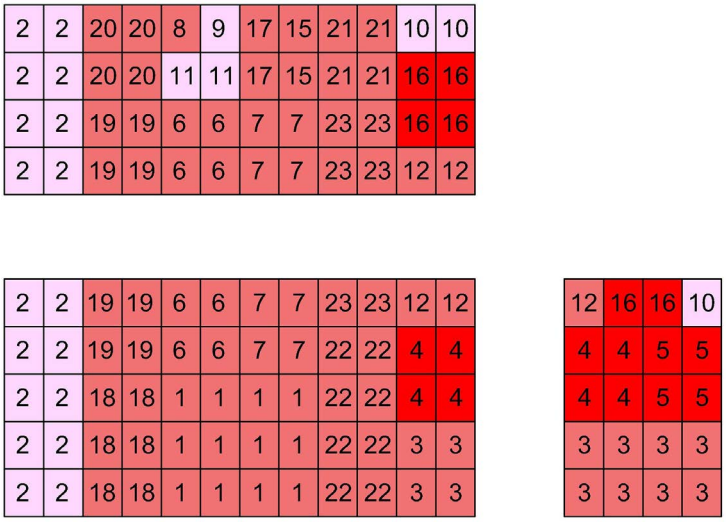

Figure 8. Priority shipment placement for optimal solution.

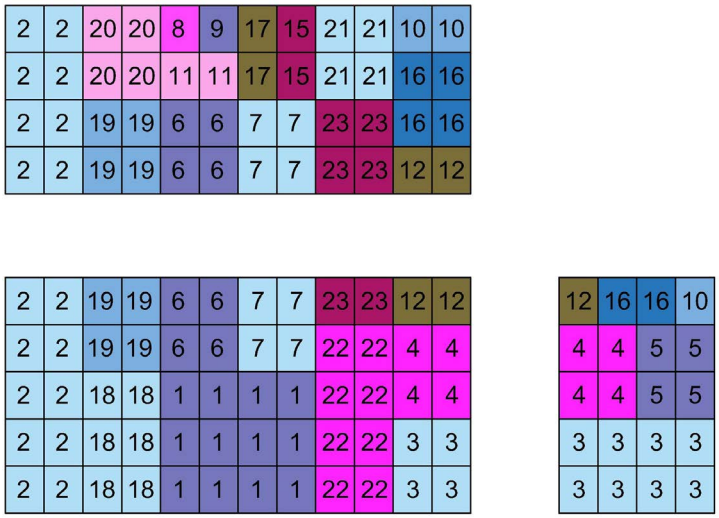

Figure 9. Front load shipment loading for optimal solution.

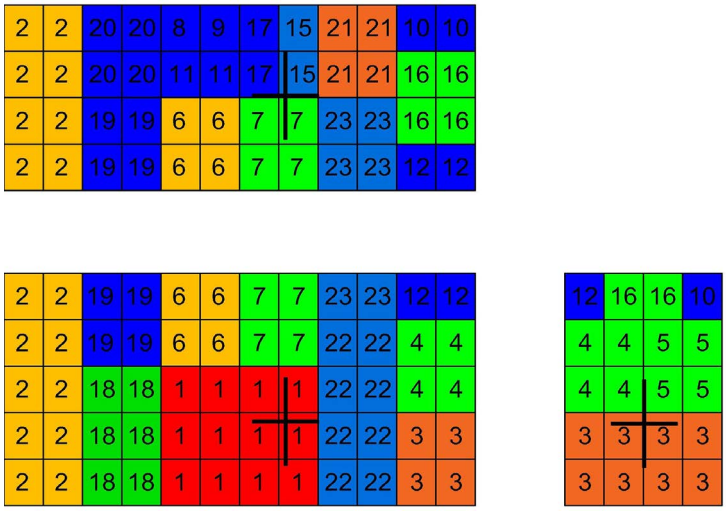

Figure 10. Center of gravity location for optimal solution.

loads are not in the rear half of the trailer. Most are packed as well as they could be. Figure 9 shows the frontload to represent a total of 52 individual volume elements (all to the hub represented by the color blue) which represents 21.6 percent of the total fill. Finally, the CG location is documented in Figure 9. The top and back views show the CG is located in the center of the trailer with respect to width, and is approximately one third of the height of the trailer. The CG location along the length of the trailer is slightly beyond the half-way point.

The solution depicted by Figures 6-10 demonstrate the effectiveness of the overall solution approach. The only goals unmet include the medium priority placement which could not be completely met due to the number and volume of these shipments. The front fill goal was slightly under the desired level and the CG location was slightly farther back than desired. The overall packing, however, was amazingly effective in meeting the goals of the shipping company. A traditionally accepted solution for the bin packing problem would be a 100 percent fill, but the solution offered here not only provides the fill, but also meets a host of other importance performance criteria. A trade-off may be made between the intelligence implemented at the agent itself and how much intelligence is provided to the agent through the genetic optimization algorithm. This balance may be adjusted over time as the methodology is adjusted for a specific application. Extension to multiple trailer loading is a straightforward process. A post optimal analysis of the result simply points out the need to form a head-load, carefully place the priority and top loads and balance the distribution of the weight of the various shipments. A traditional human loader, or even an intelligent agent implementation would have difficulty trading off the various objectives or goals without some computational support.

5. Summary and Conclusions

The use of a genetic or evolutionary algorithm to provide intelligent decision making for an agent was demonstrated for both a single objective and multiple-objective problems. The ability to deal with complex, high level goals or objectives has been documented. The use of a goal programming formulation provided a straightforward means of addressing goals which translate directly into an increase in operational efficiency. The solution generated for both the peg game and trailer loading examples demonstrate the potential of the methodology. The coupling of a genetic algorithm and an agent based framework for modeling opens a wide range of applications for systems level optimization. This is significant as most optimization approaches to decision support look at local improvements. They seldom considers the multiple, competing objectives which are present in virtually every "real world" application.

The peg game represents a simple but non-trivial application of the approach. It typifies a decision support problem as the ultimate success in solving the game is a result of each and every move made by the agent. At each move, the agent must consider a range of options, and the current positioning of the pegs is a result of all previous moves. In order to solve the problem, the optimizer must

provide the agent with the appropriate logic to generate the solution. A simple random approach, without any strategy will not solve the game in any reasonable time. Being able to generate the solution with a small population and within a relatively few number of generations is an indication of the promise of the approach in a broader context. The solution of the trailer loading problem with consideration of goals including not only fill but head-load percentage, shipment priority, top-load location and center of gravity location documents the potential in solving real world applications which are currently handled in a sub-optimal way through human intervention. The result is a direct reflection on the potential impact in not only operational efficiency of a hub, but on the safety of the loaded trailer. It also demonstrates that the evolutionary optimization algorithm can support an agent which has a large number of competing goals or objectives.

The use of a goal programming formulation allows for straightforward problem formulation, and the evolutionary optimization allows a global search to be conducted in a complex decision space. A post optimal analysis allows for local decision logic to be improved over time as well as distributing the intelligence between the agent and the optimizer. The ability of addressing multiple goals while solving a NP hard problem represents a significant achievement. Extensions to other decision support problems such as logistics and scheduling are currently being investigated.

References

[1] Shen, W., Hao, Q., Yoon, H.J. and Noorie, D.H. (2006) Applications of Agent-Based Systems in Intelligent Manufacturing: An Updated Review. *Advanced Engineering Informatics*, **20**, 415-431.

[2] Marik, V. and McFarlane, D. (2005) Industrial Adoption of Agent-Based Technologies. *IEEE Intelligent Systems*, **20**, 27-35. https://doi.org/10.1109/MIS.2005.11

[3] Yoo, M.J. (2002) An Industrial Application of Agents for Dynamic Planning and Scheduling. *Proceedings of the* 1*st International Joint Conference on Autonomous Agents and Multiagent Systems*: *Part* 1, Bologna, 15-19 July 2002, 264-271. https://doi.org/10.1145/544741.544804

[4] Davidsson, P., Persson, J.A. and Holmgren, J. (2007) On the Integration of Agent-Baser and Mathematical Optimization Techniques. In: Nguyen, N.T., Grzech, A., Howlett, R.J. and Jain, L.C., Eds., *Agent and Multi-Agent Systems*: *Technologies and Applications*. *KES-AMSTA* 2007. *Lecture Notes in Computer Science*, Vol. 4496, Springer, Berlin, Heidelberg, 1-10. https://doi.org/10.1007/978-3-540-72830-6_1

[5] Narzisi, G., Mysore, V. and Mishra, B. (2006) Multi-Objective Evolutionary Optimization of Agent-Based Models: An Application to Emergency Response Planning. *IASTED International Conference on Computational Intelligence*, November 2006.

[6] Calvez, B. and Hutzler, G. (2005) Automatic Tuning of Agent-Based Models Using Genetic Algorithms. In: Sichman, J.S. and Antunes, L., Eds., *Multi-Agent-Based Simulation VI*. *MABS* 2005. *Lecture Notes in Computer Science*, Vol. 3891, Springer, Berlin, Heidelberg, 41-57.

[7] Rogers, A. and von Tessin, P. (2004) Multi-Objective Calibration for Agent-Based Models. 5*th Workshop on Agent-Based Simulation*.

[8] Deshpande, S. and Cagan, J. (2004) An Agent Based Optimization Approach to

Manufacturing Process Planning. *Journal of Mechanical Design*, **126**, 46-55. https://doi.org/10.1115/1.1641186

[9] Gjerdrum, J., Shah, N. and Papageorgiou, L.G (2001) A Combined Optimization and Agent-Based Approach to Supply Chain Modelling and Performance Assessment. *Production Planning and Control*, **12**, 81-88. https://doi.org/10.1080/09537280150204013

[10] Sirikijpanichkul, A., Van Dam, K., Ferreira, L and Lukszo, Z. (2007) Optimizing the Location of Intermodal Freight Hubs: An Overview of the Agent Based Modelling Approach. *Journal of Transportation Systems Engineering and Information Technology*, **7**, 71-81.

[11] Neagu, N., Dorer, K., Greenwood, D. and Calisti, M. (2006) LS/ATN: Reporting on a Successful Agent-Based Solution for Transport Logistics Optimization. *IEEE Workshop on Distributed Intelligent Systems*: *Collective Intelligence and Its Applications*, Prague, 15-16 June 2006, 213-218. https://doi.org/10.1109/dis.2006.46

[12] Botterud, A., Mahalik, M.R., Veselka, T.D. and Ryu, H.S. (2007) Multi-Agent Simulation of Generation Expansion in Electricity Markets. *IEEE Power Engineering Society General Meeting*, Tampa, FL, 24-28 June 2007, 1-8.

[13] Rao, R.L. and Iyengar, S.S. (1994) A Stochastic Approach to the Bin-Packing Problem. *Proceedings of the* 1994 *ACM Symposium on Applied Computing*, Phoenix, Arizona, 6-8 March 1994, 261-265. https://doi.org/10.1145/326619.326742

[14] Rohlfshagen, P. and Bullinaria, J.A. (2007) A Genetic Algorithm with Exon Shuffling Crossover for Hard Bin Packing Problems. *Proceedings of the* 9*th Annual Conference on Genetic and Evolutionary Computation*, London, 7-11 July 2007, 1365-1371. https://doi.org/10.1145/1276958.1277213

[15] Pimpawat, C. and Chaiyaratana, N. (2004) Three-Dimensional Container Loading Using a Cooperative Co-Evolutionary Genetic Algorithm. *Applied Artificial Intelligence*, **18**, 581-601. https://doi.org/10.1080/08839510490483260

[16] Corcoran, A.L. and Wainwright, R.L. (1992) A Genetic Algorithm for Packing in Three Dimensions. *Proceedings of the* 1992 *ACM/SIGAPP Symposium on Applied Computing*, Kansas City, 1021-1030. https://doi.org/10.1145/130069.130126

[17] Mantean, O. (2007) Genetically Designed Heuristics for the Bin Packing Problem. *Proceedings of the* 9*th Annual Conference Companion on Genetic and Evolutionary Computation*, London, 7-11 July 2007, 2869-2872. https://doi.org/10.1145/1274000.1274088

[18] Lau, H. C. and Wang, H. (2005) A Multi-Agent Approach for Solving Optimization Problems Involving Expensive Resources. *Proceedings of the* 2005 *ACM Symposium on Applied Computing*, Santa Fe, New Mexico, 13-17 March 2005, 79-83. https://doi.org/10.1145/1066677.1066699

[19] Epstein, L. and Van Stee, R. (2006) This Side Up! *ACM Transactions on Algorithms*, **2**, 228-243. https://doi.org/10.1145/1150334.1150339

[20] Lim, A. and Zhang, X. (2005) The Container Loading Problem. *Proceedings of the* 2005 *ACM Symposium on Applied Computing*, Santa Fe, New Mexico, 13-17 March 2005, 913-917. https://doi.org/10.1145/1066677.1066888