Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

MULTI-AGENT RESOURCE ALLOCATION (MARA) FOR MODELING CONSTRUCTION PROCESSES

Yang Liu
Yasser Mohamed

3-133 Markin/CNRL Natural Resources Engineering Facility
Dept. of Civil and Environmental Engineering, University of Alberta
Edmonton, AB, T6G2W2, CANADA

ABSTRACT

Multi-Agent Resource Allocation (MARA) is a field developing solutions to the problem of distributing a number of resources amongst multiple agents. This field has interdisciplinary characteristics and relates to a wide range of applications, such as industrial procurement, scheduling and network routing. Many construction operations involve entities sharing and competing for limited resources. The decision to allocate these resources to entities usually has a significant impact on the schedule and cost of these operations. The dynamic and continuously changing nature of construction operations justifies the need for decision support tools with high adaptability and handling of uncertainty which is featured by MARA. This paper presents the main elements and techniques in MARA and discusses a sample case applying these techniques for the modeling of industrial construction assembly processes, also presents the conceptual model of the sample case and a prototype implementation of that model using Repast multi-agent simulation package.

1 INTRODUCTION

Multi-Agent Resource Allocation (MARA) manages the distribution of multiple resources amongst agents (Chevaleyre 2005; Chevaleyre et al. 2005). MARA has developed rapidly in recent years due to the growth of computer technology, and is gaining more and more attention because of its ability to model interactions between multiple agents and resources, a feature which makes it more industry-oriented than traditional allocation methods. In specific, MARA can facilitate the allocation of multiple resources to multiple agents. For this reason, agent-based simulation approaches, including MARA, have been applied to numerous areas, such as industrial procurement, manufacturing, network routing, public transportation, logistics (Chevaleyre 2005), electronic commerce—such as B2B (business to business) and C2B (customer to business)—(Chen et al. 2008), social activities (Furtado et al. 2007), scheduling (Liu et al. 2007), and manufacturing grid resource allocation (Yufeng et al. 2008). Although MARA has not been readily applied in construction, multi-agent system (MAS) has been recently used in project management sub-fields such as supply chain coordination (Xue et al. 2007), change order negotiation (Ren et al. 1 May 2002), equipment management (Tatari and Skibniewski 2006), and durability assessment (Ugwu et al. 2005). According to Vidal (2006), the objective of MAS is to build a complex system composed of autonomous agents capable of reaching their own goals without intervention from the user. Compared to MAS, the problem for MARA is confined to resource allocation; however, it is more detailed and formulated. Furthermore, MAS emphasizes agent autonomy while MARA is more concentrated on the agents' social welfare.

MARA provides a potential solution to those problems for which the production environment is complex and subjected to uncertain changes, which is frequently the case in construction area. This paper presents the main components of MARA and the manner in which a problem can be modeled using these components. It also discusses some alternative approaches for representing these components and some available platforms by which to execute them. Afterwards, the paper discusses implementation of MARA in the construction domain with specific reference to a sample case and the potential development of this case.

1.1 Components of MARA

1.1.1 Agents

In the artificial intelligence context, agents are defined as "[a] computational system that is situated in a dynamic environment and is capable of exhibiting autonomous and intelligent behaviour" (Russell and Norvig 2003). Agents in MARA models inherit this autonomy, which means they can apply corresponding actions which allow them to reach their goals or maximize their benefits.

Traditional object oriented programming or simulation uses objects as entities to perform tasks. Objects lack the intelligence to exploit unexpected alternatives to reach a goal or status because they only follow the instructions provider either by human users or other objects. If other objects or users fail to provide strategies in time they cannot update it simultaneously.

Autonomy is a key characteristic of agent-based systems. An autonomous agent can perform tasks with its own engine or intelligence to solve a problem without external interference—either from human users or other agents. It is capable of responding to different situations and applying alternative strategies to reach certain goals (Jennings et al. 2000). Autonomy, however, may lead to conflicts between the interests of different agents. In such cases, a special agent may be assigned the responsibility of coordinating and resolving these contradictions.

1.1.2 Resources

In the context of MARA, resources refer to the items waiting to be allocated. Resources can be categorized into two types: continuous and discrete. For continuous resources such as electricity, multiple agents can share one resource simultaneously; discrete resources, such as crews, are indivisible, such that once the resource is allocated to one agent, other agents cannot utilize the same resource.

Resource type is also distinguished in terms of its behaviour in relation to time. Resources which do not change their properties during the allocation phase are called static resources, and those which do change their properties are notated as non-static. In the majority of cases, resources are non-static, as at any given moment there are likely to be changes occurring with respect either to their numbers or to other properties. Resource type can significantly impact the allocation procedure afterwards.

1.1.3 Agent Preferences

Agent preference represents the agent's degree of satisfaction by certain allocation. Each agent has a preference value expressed as an explicit value or a relationship revealing the most satisfying alternative. An allocation procedure attempts to provide agents with alternatives of resources which, as much as possible, match their preferences.

Agent preference can take on any of a number of structures, including cardinal, ordinal, binary, and fuzzy. Cardinal preferences are to be distinguished from other structures as they always have an evaluation for the preference, either in quantitative or qualitative form. This evaluation for the preference is generally expressed as a utility function, which an agent uses to convey its degree of satisfaction with alternative combinations of resources allocated to it. The math expression here would be u: x→ val, where *u* represents the utility function for a specific agent, *x* stands for a set of allocation alternatives, and *val* refers either to a numeric value or to a linguistic term.

Ordinal preference structure does not have a utility function; however, it has a binary order between every two alternatives. This order demonstrates which alternative is the better alternative for the given agent, or at least that the two alternative expressed in the binary can be regarded as equal (Chevaleyre 2006).

As the binary and fuzzy structures cannot be as readily applied as the first two, we only address cardinal preference structure, which is later applied in our case study.

1.1.3.1 Bundle Enumeration

For a set of resources, *R*, each resource in this set will correspond to a utility function . This is referred to as explicit form because all utility functions are calculated such that as the number of resources increases, the number of utility functions rises exponentially. For the allocation procedure, it becomes quite time-consuming to compare between all these functions.

1.1.3.2 K-additive Function

In order to simplify this bundle enumeration form, researchers have developed a succinct approach in order to express utility functions as k-additive functions. A k-additive approach allows agents to have only one function which combines all the resources and can still calculate each utility when they hold different resources (Chevaleyre 2006).

1.1.3.3 Weighted Propositional Formulas

Weighted propositional is an approach typically employed to resolve a conflict between individual decisions within a small group. A weight is given to express satisfaction of an agent with the various members of a specific field of alternatives. A disutility function is calculated to represent the agent's dissatisfaction with the alternative. The best alternative for the group will be the one which has a minimum collective disutility function. For this case study, we refer to Lafage and Lang's work (2000).

Based on the above description, it is obvious that in the weighted propositional formula, part of the allocation procedure is integrated into the agent preference, which can be considered as a pre-allocation. This method is suitable for small-scale group decision making.

1.1.3.4 Bidding or Auction languages

The practice of bidding on combinations of items rather than on a single item is carried out in combinatorial auc-

tions (CA), which have widely used in algorithm mechanism design and other areas (Cramton et al. 2006).

In MARA models, auction mechanisms are used to implement the representation of agent preference. An agent offers a price to auctioneer for the resource it desires; for each agent, then, we have {Bi, pi}, where *Bi* represents the bidding item and *pi* represents the price the agent it is inclined to assign. In bidding language terms, we use "exclusive or" (XOR) and "or" (OR) to represent the combinations of bidding items. XOR can refer to either Item A or B, however, OR refers to the set of {A,B}, or any element belong to the set.

In this form of preference, the price of the resource represents the extent of how much the agent desires this resource. In this sense, agents are integrated with intelligence as they choose the most desirable resources as their bids.

1.1.4 Social Welfare

Social welfare (SW) is a terminology in economics which refers to "[a]ny of a variety of governmental programs that provide assistance to those in need" (Britannica Concise Encyclopaedia). In the context of MARA in particular, social welfare represents the aggregation of all individuals' utility functions. These aggregations may configured in a number of ways:

- Utilitarian Social Welfare: here the aggregation is calculated by the summation of all individuals' utility functions, $sw_u(p) = \sum u_i(p)$
- Egalitarian Social Welfare: this form is calculated by the agent using the minimum utility function; as Chevaleyre et al. (2005) report, "[t]his CUF (Collective Utility Function) offers a level of fairness and may be a suitable performance indicator when we have to satisfy the minimum needs of a large number of customers."

Furthermore, there are a number of other aggregation systems, such as Nash Product, Elitist, and Leximin Ordering. For further information in this regard the reader is referred to "Introduction To Multi-agent Systems" by M. Wooldridge (2001).

1.1.5 Allocation Procedure

Once agent preference has been calculated, we need a system to distribute the resources according to this preference, which is precisely the purpose allocation procedures serving in MARA. It provides a search algorithm to ensure that resources are distributed according to the specified preferences, and attempts to maximize social welfare. If there is conflict between social welfare and an individuals' preferences, allocation procedure will give the social welfare priority and negotiate with the specific agent to abandon the original preference in favour of the second choice. Figure 1 illustrates the components of MARA in the bidding mechanism. First, multiple agents bid on the resources and submit their bids—including specified bid items (combination of resources) and prices (utility functions)—to an auctioneer. Then the auctioneer will apply certain algorithms in order to increase the collective utility function (social welfare), and will negotiate with any agents that have a contradiction with their original intent.

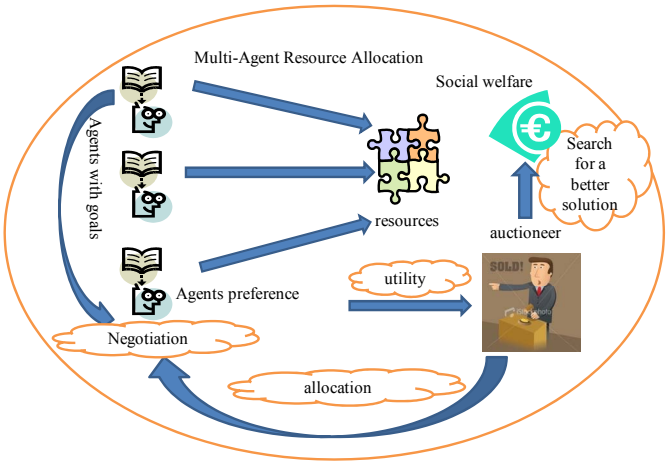

Figure 1: Components of MARA in bidding mechanism

2 APPLICATION OF MARA IN CONSTRUCTION PROCESSES

As mentioned above, MARA is an emerging system that deals with resource allocation problems. In the construction industry, labour and equipment scheduling, space allocation, and layout optimization are all problems that involve different forms of resource allocation. Generally speaking, to a degree resource allocation can be considered a constraint satisfaction problem (CSP). CSPs consist of variables, constraint sets, and objective functions. Variable values which satisfy the goal and which do not violate the constraints are called "legal assignments" (Russell and Norvig 2003). CSP has a remarkably wide range of applications in various industries, from AI planning (Gent et al. 2008; Tounsi and Ouis) to dynamic scheduling (BenHassine and Ho 2007).

One of the most distinguishing characteristics of construction operations is the considerably high level of uncertainty involved, which may be attributable to design, client, weather, labour, material, and a number of other sources. Any changes which occur in a predecessor process could lead to unexpected consequences in its successor processes, but the multi-agent modeling approach has the potential to deal with this uncertain and dynamic nature of construction operations. The autonomy of agents

causes every agent to seek its best alternative without influencing other agents. The following section describes the formulation of a MARA model for the process of assembling pipe spool modules.

2.1 Sample Case: Module Assembly

On industrial construction projects such as oil-refining and processing, pipe spools are usually assembled in modular dimensions and then shipped to a construction site as basic construction unit. Modules are available in a number of different sizes and can be classified into different types according to their main components (e.g., equipment modules, pipe rack modules). Module type and size dictate the space and location requirements for the assembly process. An assembly yard is usually located near the pipe spool fabrication area and the process begins when all the components of a module—including steel frames, pipes, and combinations of miscellaneous parts—are ready and the assembly space, usually referred as a bay, is available. However, due to delivery changes or transportation delays, the arrival time of the components suffers enormous uncertainties. Furthermore, the process must satisfy other logical, crew availability, and spatial constraints (Mohamed et al. 2007). For instance, modules can not be shipped until the space in front of them is empty. This means if a module is completed, yet there is another module in front of it being assembled in the same bay, the previous module has to wait until the path of shipment is clear.

As a result, to the task of scheduling a module assembly process and allocating the modules to a vacant space (bays) presents a major task. Usually, schedule and allocation planning is done by experienced staff using the CPM (critical path method). However, CPM only emphasizes the variance between plan and reality, and is thus incapable of responding to changes without tedious human intervention (Wang 2006).

The emergence of simulation tools within the field of construction addresses some of the limitations of CPM. It uses both resources and logic relationships to control the construction flow and the system automatically changes subsequent steps, requiring no manual modifications in order for changes to occur. Davila Borrego (2004), for instance, has implemented simulation tools in module assembly for schedule generation using discrete event simulation (DES) under the Simphony environment.

Compared to traditional DES methods, MARA concepts provide more explicit structure in dealing with resource allocation problems. The particular way a model is formulated using these concepts allows experimentation with different components of the model separately (e.g., modeling preference, social welfare, agent utility, or allocation algorithm). This structure facilitates better understanding, testing, modification, and evaluation of modeling alternatives for these components.

2.2 Model Formulation

MARA concepts are used to model the module assembly problem described in the previous section. Modules with different types, ship dates, assembly durations, and units are waiting to be assembled in multiple bays, which also have their own space units, respective types, and logical constraints. The objective is to generate a schedule that allows all modules to be shipped on or before their due date, or at least to minimize their delay.

The source of data is based on Davila Borrego's work (2004), and all data are uploaded from a database, including properties of modules, bays, their logical constraints, and the crew's types, sizes, performance factors, and so on. In this sample case, we consider spool modules as agents and assembly bays and crews as resources. We apply the widely used bidding language representing the agents' preferences.

Figure 2 shows the main elements of this sample case. Module agents express their preference to different bays which match their requirements, choose bays which offer a higher utility for them, and submit their bids for these bays to the auctioneer, The consultant collects all the bids and allocates bays to modules based on bid prices, always working to maintain a high value for the CUF. If there is a conflict between the CUF and the module's alternative, the consultant negotiates with the specific module in order that module will take the second-best alternative.

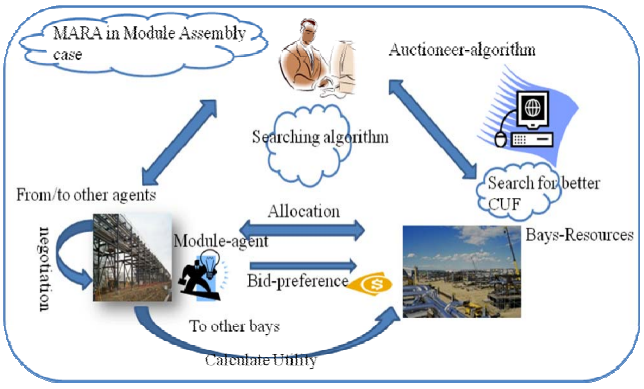

Figure 2: Module assembly case in MARA

2.2.1 Model Components

The following section describes an abstract mathematical representation of different components of the model:

Agents: Modules are represented by agents, we represent them as M (M1,M2,M3…Mn).

Resources: In our case, bays and crews are represented as resources. Bays can contain several modules at the same time based on their sizes, while the crews cannot be shared at the same time, meaning that the bays are divisible while the crews are not. We represent the bays as B(B1,B2,B3….Bn) and the Crews as C(C1,C2,C3….Cn).

Agent preference: An agent's preference (P) of an allocation to a bay (B_j) is represented by a utility function incorporating any criteria and constraints which affect the satisfaction of agents.

The utility function $u_i P(B_j)$ in our case is a combination of time factors and type factors. A module begins to calculate utility only once its planned Early Start date (ES) has arrived. For modules with the same ES, those with less float time are assigned a higher priority; however, if the ES is passed and the module still has not allocated, then the utility function will be given a penalty such that the Social Welfare (SW) decreases as well.

The final utility function formula is expressed as follows:

$$u_i P(B_j) = C - [t_{due} - (ES + t_{required})] - Penalty + w_{j_{type}} \quad (1)$$

Where $w_{j_{type}}$ is a type factor, if bay j's type matches the module's type, then the factor is equal to 1; otherwise, the type factor approaches 1 as the bay's type approaches the module's type.

C is a large constant introduced to ensure that the utility function is positive.

$t_{due}, ES, t_{required}$ represent the module's ship date, ES, and assembly duration, respectively. The item $[t_{due} - (ES + t_{required})]$ represents the module's free float, and a penalty is calculated using the (ES-current time)*penalty multiplier,

Social welfare SW $_u(P) = \sum_{i \in A} u_i(P)$

Allocation algorithm: the allocation procedure is divided into the following steps:

1. As we approach the ES, we may calculate the utility function of all modules with the same ES according to their preferred route

$$M1\begin{cases} u_1P(B_1) \\ u_1P(B_2) \\ u_1P(B_3) \\ \cdots\cdots\cdots \\ u_1P(B_n) \end{cases}, \quad M2\begin{cases} u_2P(B_1) \\ u_2P(B_2) \\ u_2P(B_3) \\ \cdots\cdots\cdots \\ u_2P(B_n) \end{cases}, \ldots\ldots \quad Mn\begin{cases} u_nP(B_1) \\ u_nP(B_2) \\ u_nP(B_3) \\ \cdots\cdots\cdots \\ u_nP(B_n) \end{cases}$$

2. For each Module i, the agent selects its largest number of utility functions for different bays and assigns this specific number as a bid to the auctioneer, who will fulfill the role of an auctioneer in a bidding game.
3. The auctioneer applies the best first search to confirm the specific match (allocation) according to a descending order of the utility function value. Assuming that $U\{u_1P(B_j), u_2P(B_k) .. u_nP(B_n)\}$ is a set consisting of the greatest utility function of each module, the auctioneer will allocate the bays to the highest bidder in set U, then to the second-best one, and so on. Meanwhile, once the highest bidder is given their bay, the bay's vacancy is subtracted from the module's requested space units.

2.2.2 Execution Flow

The basic flow in this model starts from the site manager, which represents the model class. It prepares the bays and modules and executes distribution commands in every time tick. For each module, once it has reached its ES, it calculates its utility function for all the bays, selecting the highest as a bid item and its utility function as a bid price. A valid bid includes the module agent that placed the bid, the bay requested, and the bid price. Then the site manager commands modules to report all the bids to an auctioneer. Once the auctioneer has acquired all the agents' bids, it uses the allocation algorithm to decide which module is to be served first. Figure 3 summarizes the basic flow. Details related to bay and module attributes and the changes in their values after allocation are not shown in the figure.

3 MODEL IMPLEMENTATION

There are several commercial and open source agent-based simulation platforms available, such as AnyLogic, Swarm, NetLogo, and Repast. Repast.NET is used as the framework to implement MARA for this sample case.

Repast is an abbreviation for Recursive Porous Agent Simulation Toolkit. It borrows some concepts from Swarm, but it also supports multiple languages, java, .net, c# and python script so that the user can build the model in any of these languages. Also, although Repast is a time stepped simulation platform, the user can set it to execute in much the same manner as DES. If there are no behaviours occurring in some time ticks, the system may skip over them.

Figure 4 shows a screen shot of the model in sixth time tick from Repast3.Net, where the background is the assembly yard layout. Modules which have already been allocated are represented by blue bars, and those that have not yet been allocated are represented by red bars. Once the assembly process has been completed, the module disappears from the bay in the display interface.

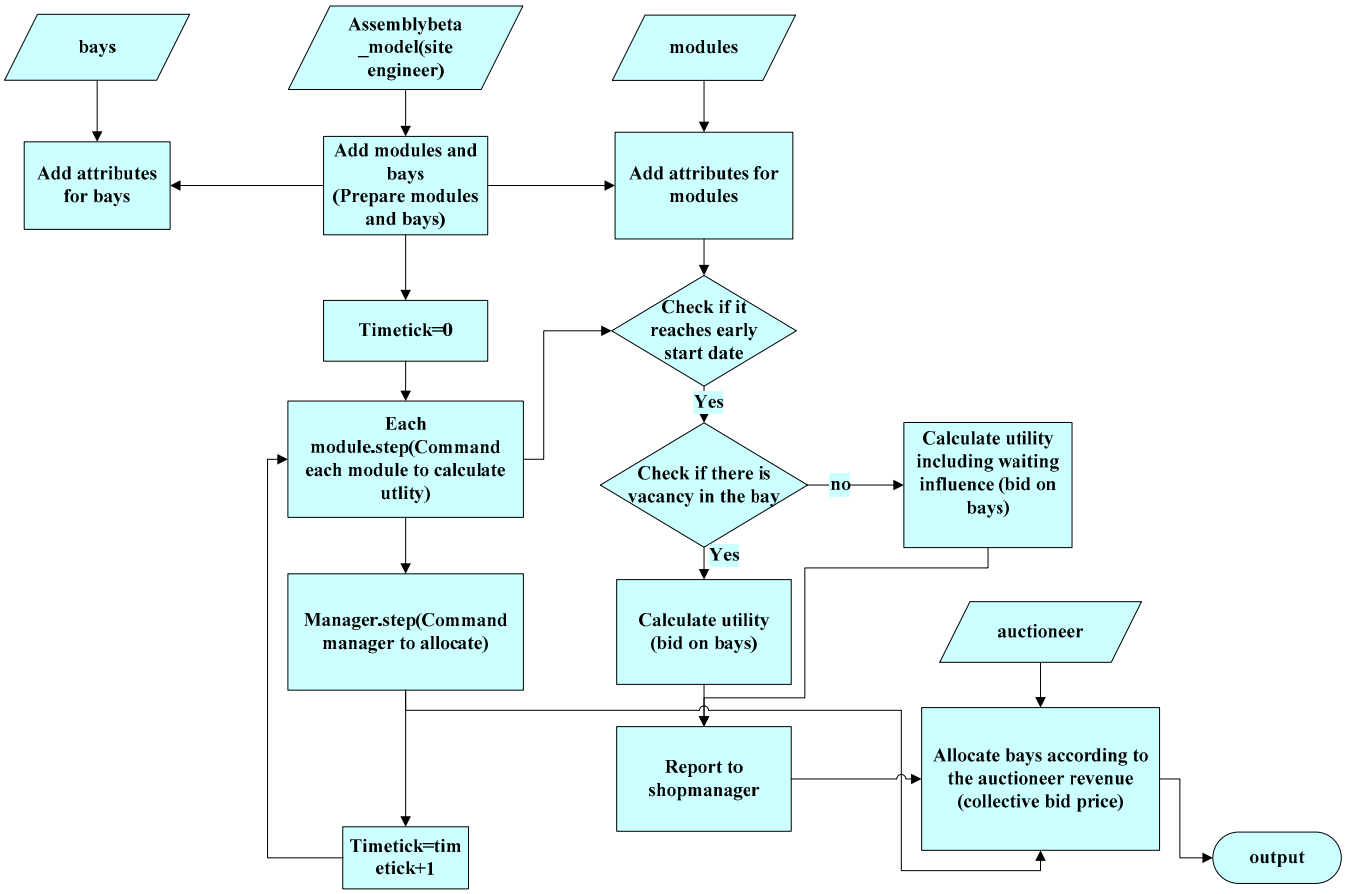

Figure 3: Basic flow of model

Figure 5 shows the change in social welfare calculated by the auctioneer agent during the allocation process between time tick 0 and 170. Social Welfare (or CUF) is a summation of all individuals' utility functions; by the time a module is allocated, its utility has been added to the CUF.

To reflect real world constraints such as the limitations of crane coverage area and shipment blocking further behaviours are added to the model as follows:

- Instead of submitting only the bid with the highest price. modules submit several bids corresponding bays, and are stored in the bay's request list.
- A block penalty is taken into consideration in the utility function and updated in a at each time tick.
- In addition to best-first search, the auctioneer can employ dynamic programming algorithm to allocate modules without changing the other components of model. Auctioneer applies 0-1 knapsack (Corner 1991, and Cormen 2001) strategy to maximize the collective price of modules in a bay's request list.

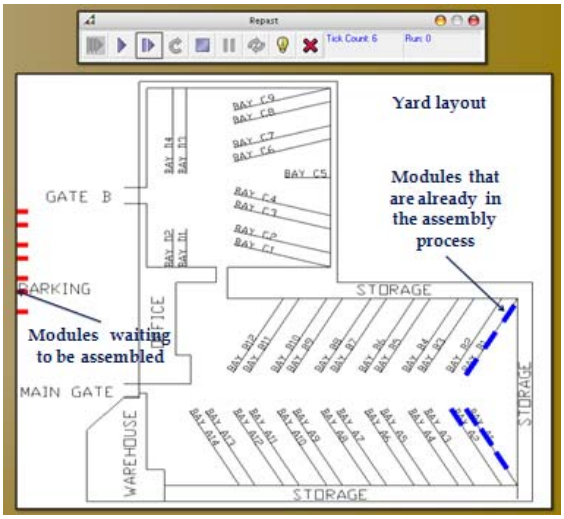

Figure 4: Agent display of 6th time tick

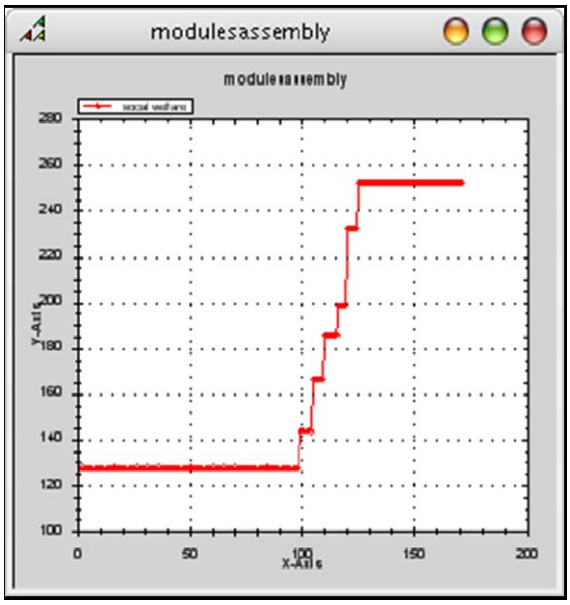

Figure 5: Social welfare of during time tick 0 to 170

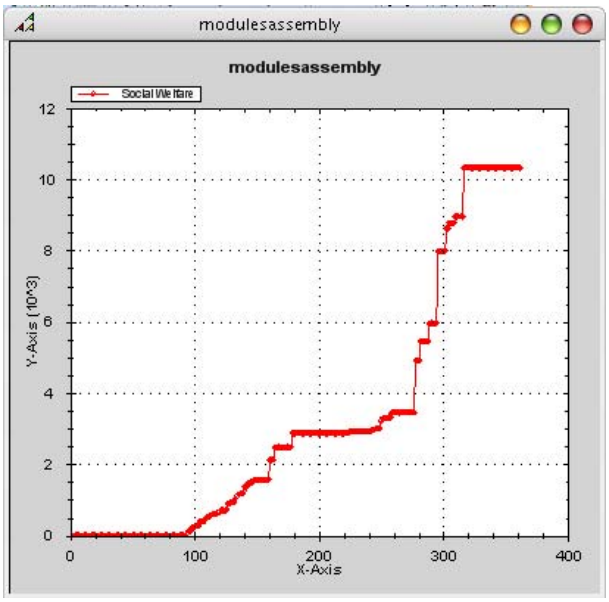

Figure 6: SW, BST=0, BSTP=10, Best-first search

Figure 6 and 7 illustrates the difference of Social Welfare between two allocation algorithms under the circumstance that Block Ship Tolerance (BST) is 0, and Block Ship Penalty (BSTP) is 10.

BST represents the threshold of the maximum delay when a module is blocked by another module in the same bay from shipment. A BST of 0 means no modules are allowed to be allocated in a bay where it will cause any delay of shipment.

BSTP represents the penalty multiplier when such delay happens. In the bidding process, utility function is decreased by BSTP times the block days. In allocation process, the auctioneer will update the utility function after each module is facilitated so that the Social Welfare can reflect the overall satisfaction of module agents.

In Figure 6 and 7, when the BST is 0, allocation procedure barely affect the result except the slight difference in terms of SW because both of the best first search and dynamic programming employ the same heuristic rule, which is the closer to a module's due date, the higher priority it has. However, dynamic programming algorithm leads to a better SW due to its nature of optimality

It was also found that as the number of modules increases, the advantage of the dynamic programming algorithm is more evident. Figure 7 and 8 are the screenshots of SW when the number of modules increase to three times as before, BST=10 and BSTP=50.

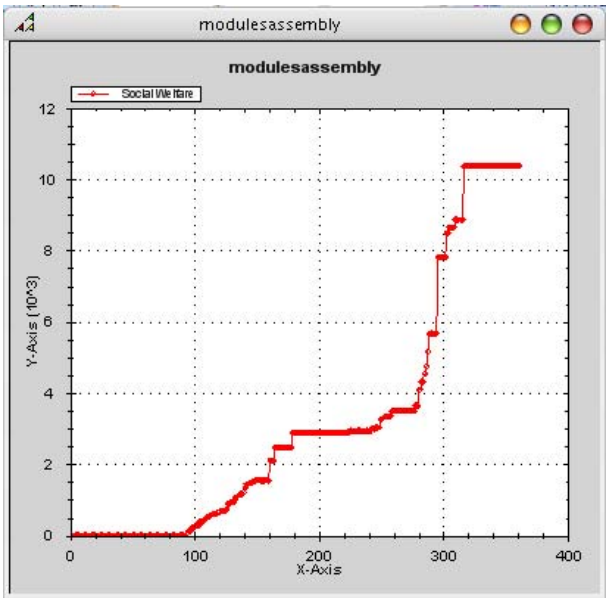

Figure 7: SW, BST=0, BSTP=10, Dynamic Programming

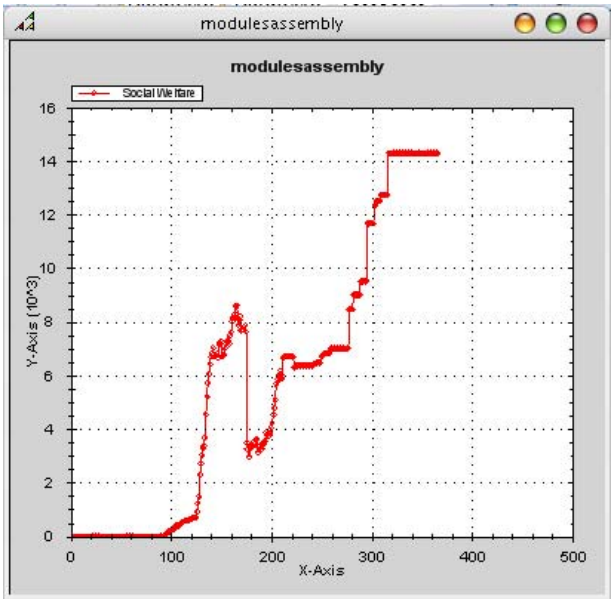

Figure 8: SW, BST=10, BSTP=50, Modules NO.=497, Best-First Search

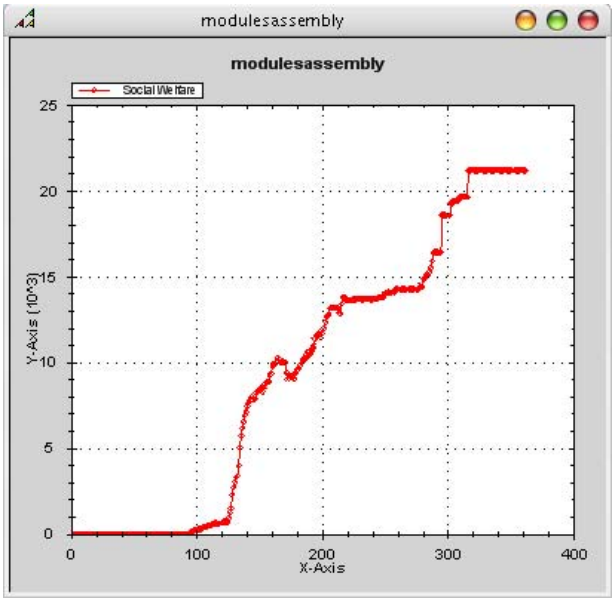

Figure 9: SW, BST=10, BSTP=50, Modules NO.=497, Dynamic Programming Search

4 CONCLUSIONS AND FURTHER RESEARCH

This paper has presented an overview of the main components of MARA models. A sample case is also presented to illustrate the use of MARA for modeling problems in the industrial construction domain. Modeling construction resource allocation problems using MARA provides a more structured approach by which to analyze those problems which present a high degree of uncertainty and dynamism associated. The conceptual model of the sample case has been presented in addition to a prototype implementation in Repast3.NET.

Two different allocation algorithm and the result of SW are discussed, and show the potential of improving the efficiency of resource distribution by using advance search methods.

However, because of the complexity of allocation between multi agent and resources, and the fact that the allocation is dynamically committed at different time spans, the dynamic programming algorithm still does not guarantee absolute optimum results, which justifies the need of future research regarding to search optimality. Other aspects related to agent learning and negotiation are proposed for future study as well.

REFERENCES

Anumba, C. J., Ugwu, O. O., and Ren, Z. 2005. *Agents and multi-agent systems in construction*, Taylor & Francis, London; New York.

BenHassine, A., and Ho, T. B. 2007. An agent-based approach to solve dynamic meeting scheduling problems with preferences. *Engineering Applications of Artificial Intelligence*, 20(6): 857-873.

Chen, D., Jeng, B., Lee, W., and Chuang, C. 2008. An agent-based model for consumer-to-business electronic commerce. *Expert Systems with Applications*, 34(1): 469-481.

Chevaleyre, Y. 2006. Multi-agent resource allocation in k-additive domains: preference represention and complexity. *Institute for Logic, Language and Computation* (ILLC), University of Amsterdam, Amsterdam.

Chevaleyre, Y. 2005. Issues in multi-agent resource allocation. *Institute for Logic, Language and Computation* (ILLC), University of Amsterdam, Amsterdam.

Chevaleyre, Y., Dunne, P., Endriss, U., Lang, J., Maudet, N., and Guez-Aguilar, J. A. 2005. Multi-agent Resource Allocation. *Knowledge Engineering Review: Special Issue on the AgentLink~III Technical Forums*, 20(2): 143-149.

Corner, J. L., and Kirkwood, C. W. 1991. Decision Analysis Applications in the Operations Research Literature, 1970-1989. *Oper.Res.*, 39(2): 206-219.

Cormen, T. H. 2001. *Introduction to algorithms*. MIT Press, Cambridge, Mass.

Cramton, P. C., Shoham, Y., and Steinberg, R. 2006. *Combinatorial auctions*. MIT Press, Cambridge, Mass.

Davila Borrego, L. F. 2004. Simulation-based scheduling of module assembly yards with logical and physical

constraints. Ph.D. thesis, University of Alberta, Canada.

Furtado, V., Melo, A., Coelho, A., and Menezes, R. 2007. A crime simulation model based on social networks and swarm intelligence. Published in the Proceedings of the ACM Symposium on Applied Computing, 56-57.

Gent, I. P., Nightingale, P., Rowley, A., and Stergiou, K. 2008. Solving quantified constraint satisfaction problems. *Artificial Intelligence*, 172(6-7): 738-771.

Jennings, N. R., Norman, T. J., Faratin, P., O'Brien, P., and Odgers, B. 2000. Autonomous Agents for Business Process Management. *Application of Artificial Intelligence*, 14(2): 145-189.

Vidal, J. M. 2006. Fundamentals of Multi-agent Systems: Using NetLogo Models. Unpublished.

Kumar, V.1992. Algorithms for constraint-satisfaction problems: a survey. *Artificial Intelligence Magazine*, 13(1): 32-44.

Lafage, C. and Lang, J. 2000. Logical representation of preferences for group decision making. Published in the *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*: 457-468.

Liu, P., Lin, Y., Wu, J., and Kang, Z. 2007. An optimal scheduling algorithm for an agent-based multicast strategy on irregular networks. *Journal of Supercomputing*, 42(3): 283-302.

Mohamed, Y., Borrego, D., Francisco, L., Al-Hussein, M., AbouRizk, S., and Hermann, U. 2007. Simulation-based scheduling of module assembly yards: case study. *Engineering, Construction and Architectural Management*, 14(3): 293-311.

Nisan, N. 2000. Bidding and allocation in combinatorial auctions. EC '00: Published in the *Proceedings of the 2nd Association for Computing Machinery (ACM) conference on Electronic commerce, ACM*, New York, NY, USA, 1-12.

Ren, Z., and Anumba, C. J. 2004. Multi-agent systems in construction–state of the art and prospects. *Automation in Construction*, 13(3): 421-434.

Ren, Z., J. Chimay, O. Anumba, and O. Ugwu. 1 May 2002. Negotiation in a multi-agent system for construction claims negotiation. *Application of Artificial Intelligence*, 16(36): 359-394.

Repast Agent Simulation Toolkit 2007. Retrieved [April, 29, 2007] from <http://repast.sourceforge.net/>

Russell, S. J. and Norvig, P. 2003. *Artificial intelligence: a modern approach*. Prentice Hall/Pearson Education, Upper Saddle River, N.J.

Tatari, O. and Skibniewski, M. 2006. Integrated Agent-Based Construction Equipment Management: Conceptual Design. *Journal of Civil Engineering & Management*, 12(3): 231-236.

Tounsi, M. and Ouis, S. A Local Search Framework for Solving Constraint Satisfaction Problem. *Applied Soft Computing*, In Press, Accepted Manuscript 771.

Ugwu, O. O., Kumaraswamy, M. M., Kung, F., and Ng, S. T. 2005. Object-oriented framework for durability assessment and life cycle costing of highway bridges. *Automation in Construction*, 14(5): 611-632.

Wang, P. 2006. Production-based large scale construction simulation modeling. Ph.D. thesis, University of Alberta, Canada.

Wooldridge, M. J. 2001. *Introduction to Multi-agent Systems*. John Wiley & Sons, Inc., New York, NY, USA.

Xue, X., Wang, Y., and Shen, Q. 2007. Agent based multi-attribute negotiation for large-scale construction project supply chain coordination. Published in the *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States, Hong Kong, China, 531-534.

Yufeng, D., Fei, T., Buyun, S., and Zude, Z. (2008). Modelling and application of optimal-selection evaluation for manufacturing grid resource. *International Journal of Computer-Integrated Manufacturing*, 21(1): 62-72.

Zurawski, R. 2005. *The industrial information technology handbook*, CRC Press, Boca Raton

AUTHOR BIOGRAPHIES

YANG LIU is an M.Sc. student in the Department of Civil and Environmental Engineering at the University of Alberta. He received his B.Sc. in Civil Engineering from Beijing University of Technique, China. His e-mail address is <liu4@ualberta.ca>

YASSER MOHAMED is an Assistant Professor in the Department of Civil and Environmental Engineering at the University of Alberta. He received his B.Sc. and M.Sc. in Civil Engineering from Zagazig University, Egypt. He received his Ph.D. degree from the University of Alberta in 2002. His research interests have focused on decision support for construction management using simulation, artificial intelligence, and knowledge engineering techniques. His e-mail address is <yaly@ualberta.ca>