2009 Sixth International Conference on Information Technology: New Generations

Model-Based Specification of Flexible and Complex Bidding Strategies in Agent-Based Online Auctions*

Benjamin J. Ford†, Haiping Xu†, Christopher K. Bates†, and Sol M. Shatz‡
†*Computer and Information Science Department*
University of Massachusetts Dartmouth, North Dartmouth, MA 02747, USA
E-mail: {u_bford, hxu, u_cbates}@umassd.edu
‡*Computer Science Department*
University of Illinois at Chicago, Chicago, IL 60607, USA
E-mail: shatz@uic.edu

Abstract

Current implementations of agent-based online auction systems only support simple predefined bidding strategies for bidding agents. In this paper, we introduce a formal bidding strategy model for specification of flexible and complex bidding strategies. The formal model is defined as a layered bidding strategy model (LBSM), which can be represented using notations borrowed from UML activity diagrams. To support real-time and efficient reasoning, the formal model is converted into a rule-based bidding strategy model (RBSM) specified in bidding strategy language (BSL) that can be directly executed by a reasoning module of a bidding agent. We present an algorithm for converting an LBSM to an RBSM, and an algorithm to drive the reasoning engine. Finally, we develop a prototype agent-based online auction system using JADE, and illustrate how flexible and complex bidding strategies can be precisely specified and efficiently executed.

1. Introduction

Online auction houses, such as eBay, have seen an increasing amount of transactions since their debut. As the number of transactions increases, researchers have been investigating the mechanisms and benefits of automating online auction activities. One major form of such automation is agent-based online auctions that are Internet-based auctions running partially or entirely through the use of software agents, where software agents can act on behalf of human users, such as buyers, sellers, and auction house administrators in online auction systems [1-4].

In an agent-based online auction system, a bidding agent can automatically place bids on behalf of a human user according to a user-specified bidding strategy. A bidding strategy consists of a set of bidding activities and conditions. During an online auction, when certain conditions become true, appropriate bidding activities (e.g., increasing the bid amount or placing a bid) can be automatically performed by a bidding agent. While there have been previous efforts on designing optimal bidding strategies [5-7], work on specifying bidding strategies for bidding agents is more rare. Current implementations of bidding agents only support simple predefined bidding strategies [8]. One other strategy specification framework utilizes a logic-based approach [9]; however, that approach lacks the flexibility necessary for specifying large and complex strategies. In order to support user-specified bidding strategies, there is a pressing need for a feasible way that allows a user to specify bidding strategies that effectively represent the user's bidding plans for performing auction activities.

In this paper, we introduce a model-based approach that supports specification of flexible and complex bidding strategies for bidding agents. Our approach divides a complex strategy into various modular layers. Simple strategies at lower layers can be assimilated into a larger and more complex strategy at a higher layer. For real-time and efficient reasoning, the formal model is converted into a rule-based bidding strategy model specified in bidding strategy language (BSL) that we propose in this paper, thus the rule-based strategy model can be directly executed by a reasoning module of a bidding agent using a reasoning engine.

* This material is based upon work supported by the U.S. National Science Foundation under grant numbers CNS-0715648 and CNS-0715657.

978-0-7695-3596-8 2009

DOI 10.1109/ITNG.2009.29

2. Related work

Previous related work includes research on designing good bidding strategies for agent-based online auctions, and work on formal specification of bidding strategies. Park et al. develop an adaptive agent bidding strategy, called the *p*-strategy, based on stochastic modeling for dynamic, evolving multi-agent auctions [5]. The *p*-strategy considers the dynamics and resulting uncertainties of an auction process using stochastic modeling, which can adaptively decide when the model should be used. Ma and Leung present the design and analysis of a new strategy for buyer and seller agents participating in agent-based continuous double auctions or CDAs [7]. The proposed strategy employs heuristic rules and reasoning mechanisms based on a two-level adaptive bid-determination method, which allows bidding agents to dynamically adjust their behaviors in response to changes in the supply-demand relation of the market. Although the above proposed bidding strategies may provide better chances for a user to win auctions, they are either not feasible for use by inexperienced and ordinary users, or they must be predefined as bidding strategies for bidding agents. In the latter case, users may not be allowed to modify or improve the bidding strategy to meet their personal preferences and needs. In contrast, our approach provides users the mechanisms to adopt an existing bidding strategy, design their own strategies, and compose available strategies into a complex one. With such mechanisms, a bidding agent can truly place bids on behalf of a human user in order to meet the user's bidding requirements.

Very little work has been done on formal specification of bidding strategies. Gimenez-funes et al. introduce both a formal and a more pragmatic approach for the design of bidding strategies with useful heuristic guidelines for buyer agents [10]. The proposed approach utilizes global and individual probabilistic information such that the resulting bidding strategy can balance the agent's short-term benefits with its long-term ones. Other research has described how defeasible logic can be utilized to specify negotiation strategies [11, 9]. Defeasible logic – although it is formal and allows users to specify rules based on uncertainty – has an inherently large learning curve due to its mathematical foundations. Efforts have been made to counter this disadvantage by utilizing digraphs to visualize the defeasible logic rules [12], but that representation still requires users to learn a notation that is not widely used.

Unlike the above approaches, in our formal bidding strategy model, we borrow some notations from UML activity diagrams [13, 14] to explicitly display strategy transitions and action transitions as a workflow of activities. Such a representation can support an easy-to-use interface for users to specify bidding strategies graphically. As a result, our approach has the potential for making it significantly easier to learn how to specify strategies, while still allowing users to specify complex and flexible bidding strategies. For example, a complex and flexible strategy may adapt to other bidding agents' bidding behaviors, possibly by examining increases in bidding increment or increases in bidding frequency for a given period of time.

3. Bidding agent architecture

Figure 1 shows an overview of an agent-based online auction system. There is a central auction house that consists of various auction agents, each of which manages an auction in progress. The bidding agents work on behalf of human users. A user who wants to bid on a particular auction must provide the bidding agent with a bidding strategy. The bidding agent then communicates with the corresponding auction agent to query for related information, such as the current highest bid and the number of active bidders in that auction. Based on the available information, the bidding agent makes a decision, and may place a bid by sending a bid request to the auction agent.

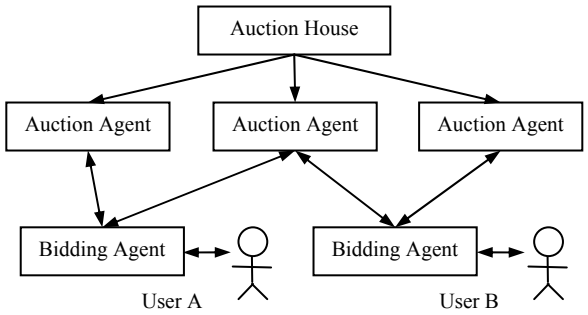

Figure 1. Agent-based online auction system

A bidding agent consists of a bidding agent interface and a reasoning module. The bidding agent interface is responsible for communicating with auction agents as well as human users. The reasoning module is used to make decisions for choosing the next bidding activity according to bidding strategies specified by a human user. Figure 2 describes the bidding agent architecture. A user can specify a layered bidding strategy model (LBSM) through the bidding agent interface. The LBSM is represented as a visual strategy model that is internally stored as an XML file. Once a strategy has been defined, it is converted into a rule-based bidding strategy model (RBSM) consisting of a set of production rules. The production rules can

be directly executed by the reasoning module for decision making. Based on the current state of the auction, the reasoning module determines the next action and sends that action to the bidding agent interface for further processing.

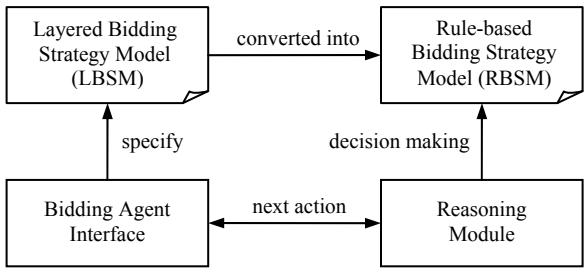

Figure 2. Bidding agent architecture

4. Layered bidding strategy model

In our model-based approach for bidding agents, we utilize a layered architecture to specify bidding strategies. A layered architecture allows specification of strategies at different levels of complexity. Figure 3 illustrates the general architecture of our layered bidding strategy model (LBSM). An LBSM consists of three layers, namely the complex-strategy layer, the simple-strategy layer, and the bidding-action layer. The complex-strategy layer defines complex strategies using strategies from the same layer and strategies from the simple-strategy layer. The functionality of switching between strategies is defined by a strategy transition, which consists of a *start* strategy, an *end* strategy and a transition condition. The simple-strategy layer defines simple strategies using bidding actions defined in the bidding-action layer. The functionality of switching between bidding actions is defined by an action transition, which consists of a *start* action, an *end* action and a transition condition. The bidding actions layer defines the atomic actions available to a bidding agent. Examples of such atomic actions include placing a bid, changing bid limit, and random pausing.

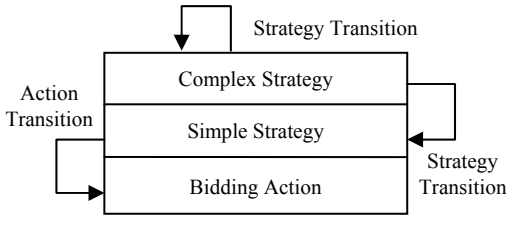

Figure 3. Layered bidding strategy model (LBSM)

Figure 4 shows an example of simple strategy called *S*1 using notations of UML activity diagrams. The initial action is a *ChangeDynamicBidIncrement* action that must be executed first whenever *S*1 is selected to execute. This initial action changes a user's bid increment to $10. The next action is a *DynamicBidAction* that places a single bid in the amount of the current highest bid plus 10 dollars. The strategy then requires a pause for a random time from 45 minutes (2700 seconds) to an hour (3600 seconds), followed by a check to see if the transition condition `¬highestBidder && highestBid+10 <= bidLimit` is true or not. This condition is used to check whether the bidder's last bid remains the highest one and whether placing another bid may exceed a predefined bid limit specified by the user. If the transition condition evaluates to *true*, the next bidding action will again be *DynamicBidAction*; otherwise, an action of *RandomPauseBidding* will be taken. This procedure must be repeated until the bid limit is reached or the auction terminates. Thus, simple strategy *S*1 represents an aggressive strategy. To simplify other diagrams presented in this paper, whenever the condition `highestBid + bidIncrement <= bidLimit` is missing for an action transition, it is assumed to be implicitly specified.

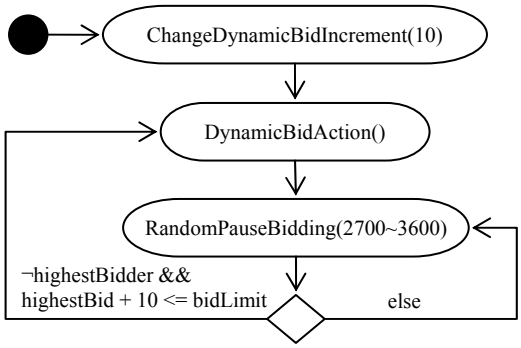

Figure 4. Simple strategy *S*1 (an aggressive one)

A complex strategy consists of a set of strategies, strategy transitions and an initial strategy. Note that a strategy in a complex strategy can either be a simple or complex one. The initial strategy is the one that is executed when the complex strategy is selected to execute. Figure 5 shows a complex strategy *C*1 that contains two simple strategies *S*1 and *S*2 (we will define simple strategy *S*2 in Section 6). *S*1 is the initial strategy of *C*1. When *C*1 is selected to execute, the initial action in *S*1 will be executed first. While the time remaining in the auction is greater than 900 seconds, *S*1 is executed continuously. Once the time remaining is less than or equal to 900 seconds, a

strategy transition must be made to simple strategy *S*2. Similarly, in this case, the initial action of *S*2 will be selected as the next action. Note that although we illustrate only two simple strategies in *C*1, a complex strategy may also contain other complex strategies as components.

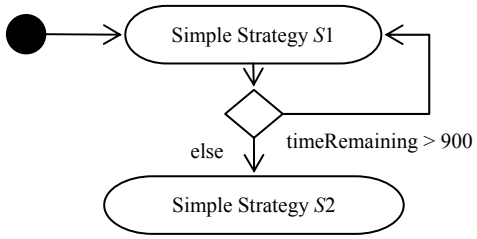

Figure 5. Complex strategy *C*1 (complex aggressive)

5. Rule-based bidding strategy model

To facilitate efficient execution of a strategy by the reasoning module, we define a formal language called bidding strategy language (BSL) to specify rule-based bidding strategy models (RBSM). A rule-based bidding strategy model can be automatically converted from an LBSM, but it allows efficient reasoning, and may potentially support being expanded with new rules enforced by an auction house in real-time. Figure 6 gives the formal definitions of BSL in Backus-Naur Form (BNF). From the definitions, we can see that a rule-based bidding strategy model is specified as production rules that can be strategy rules, action rules, initial strategy rules or initial action rules. A strategy rule corresponds to a strategy transition with an *s-domain*, which specifies the strategy transition's enclosing strategies at different levels. For example, if complex strategy *C*1 contains simple strategies *S*1 and *S*2, and if *C*1 itself is defined as a component of complex strategy *C*2, a strategy rule that transits from *S*1 to *S*2 would have an *s-domain* of *C*2.*C*1.

Similarly, an action rule corresponds to an action transition with an *a-domain*, which specifies the action transition's enclosing strategies at different levels. Note that an *a-domain* of an action rule follows the same principle as that of a strategy rule, but its closest enclosing strategy must be a simple strategy rather than a complex one. An initial strategy rule is a special rule that defines the first strategy to be used in a complex strategy; while an initial action rule defines the first action to be taken in a simple strategy. Thus, a strategy defined in BSL is essentially a set of production rules, which defines action rules and initial action rules for simple strategies, and strategy rules and initial strategy rules for complex strategies.

```
<production rule>::= <strategy rule> |
  <action rule> | <initial strategy rule> |
  <initial action rule>
<strategy rule> ::= <s-domain> <bidding
  strategy> <condition> -> <bidding
  strategy>
<s-domain> ::= <s-domain>.<complex strategy>
  | <complex strategy>
<bidding strategy> ::= <simple strategy> |
  <complex strategy>
<condition> ::= <compound condition> |
  <arithmetic condition> | <comparison
  condition> | <boolean condition>
<action rule> ::= <a-domain> <action>
  <condition>  -> <action>
<a-domain> ::= <s-domain>.<simple strategy>
  | <simple strategy>
<action> ::= <basic bid> | <change bid
  limit> | <change dynamic bid increment> |
  <dynamic bid> | … | <pause> | <stop>
<initial strategy rule> ::= <complex
  strategy>  -> <initial strategy>
<initial action rule> ::=  <simple strategy>
  -> <initial action>
```

Figure 6. Definition of BSL in Backus-Naur Form (BNF)

The model conversion algorithm (Algorithm 1) converts a user-specified LBSM into a set of rules that can be executed directly by the reasoning module. The algorithm first checks whether an LBSM describes a complex strategy. If so, it creates an initial strategy rule and a list of strategy rules based on the LBSM. Once the list of strategy rules has been created, the algorithm starts to process each strategy contained in the LBSM recursively. When the algorithm reaches its base case (i.e., LBSM is a simple strategy), it creates an initial action rule and a list of action rules.

Algorithm 1. Model Conversion

function convertToRuleBasedStrategyModel (LBSM *lbsm*)
  if *lbsm* is a complex strategy
    add a new initial strategy rule:
      lbsm → *lbsm.initialStrategy*
    for each StrategyTransition *st* in *fbsm*
      set up *s-domain* according to the strategy hierarchy
      add a new strategy rule: *s-domain, st.startStrategy,*
        st.condition → *st.endStrategy*
    end
    for each strategy *s* in *lbsm*
      convertToRuleBasedStrategyModel (*s*)
    end
  else if *lbsm* is a simple strategy /* base case */
    add a new initial action rule: *lbsm* → *lbsm.initialAction*
    for each ActionTransition *at* in *lbsm*
      set up *a-domain* according to the strategy hierarchy
      add a new action rule: *a-domain, at.startAction,*
        at.condition → *at.endAction*
    end
end function

Algorithm 2. Reasoning Engine

```
function Action findNextAction
        (Domain domain, Action currentAction)
  if currentAction == null
    if domain is a ComplexStrategy
      Search for initial strategy rule isr for domain that
      leads to initial strategy is
      return findNextAction (is, null)
    else if domain is a SimpleStrategy
      Search for initial action rule iar for domain that
      leads to initial action ia
      return ia
  else if currentAction != null
    Remove and process the first element fe of domain,
    and let the remaining domain be r-domain
    if fe is a ComplexStrategy /* strategy transition */
      Retrieve all strategy rules for the first element of
      r-domain and store them in a list
      while the list is not empty
        Remove and process strategy rule sr at list head
        if the condition for sr is true
          Let s be the conclusion part of sr
          return findNextAction (s, null)
      return findNextAction (r-domain, currentAction)
    else if fe is a SimpleStrategy /* action transition */
      Retrieve all action rules for the currentAction and
      store them in a list
      while the list is not empty
        Remove and process action rule ar at list head
        if the condition for ar is true
          return the conclusion part of ar
      return currentAction
end function
```

Algorithm 2 describes the reasoning algorithm with two parameters: *domain* and *currentAction*. The parameter *domain* refers to the strategy hierarchy of the strategy where *currentAction* happens, and *currentAction* is the last action taken by the bidding agent. For example, when *domain* is *C*2.*S*1 and *currentAction* is *a*2, it tells the reasoning module that the last action taken by the bidding agent is *a*2, which is defined in simple strategy *S*1, and *S*1 is a component of complex strategy *C*2. Note that the last element of *domain* must be a simple strategy because a bidding action cannot appear in a complex strategy. However, if *currentAction* is *null*, *domain* can refer to either a complex or a simple strategy. In either case, the initial action of *domain* is returned as the next action.

On the other hand, if *currentAction* is not *null*, the reasoning module will search from the highest level of *domain*. Any transition at a higher level of *domain* has higher priority than transitions at lower levels. For example, if *domain* is *C*2.*S*1, the reasoning module first searches in strategy *C*2 for any possible strategy transition from *S*1 (i.e., the corresponding transition condition is *true*). If such a transition is found, say *S*1 can switch to *S*2 in *C*2, the next action will be the initial action of *S*2. Otherwise, the reasoning module searches in strategy *S*1 for any possible action transition from *currentAction*. If such a transition is found, say *currentAction* can switch to *a*2 in *S*1, the next action will be *a*2. Otherwise, if all levels of *domain* have been searched and no next action can be found, *currentAction* is returned as the next action.

6. Case study

To demonstrate how bidding strategies can be easily specified and how execution of different specified strategies may influence the bidding behaviors in agent-based online auctions, we present a case study for a fictitious auction – for an item with an estimated auction price of around $1000. We first design a set of simple and complex strategies that are used in our experiments.

Figure 7 shows a simple strategy *S*2, which is a very aggressive strategy. Using *S*2, the bidding agent first changes the dynamic bid increment to $5, and also changes the bid limit to $1500. Then the bidding agent continuously places dynamic bids with an increment of $5 every 15 seconds, as long as this bidder is not the highest bidder and the bid limit has not been exceeded.

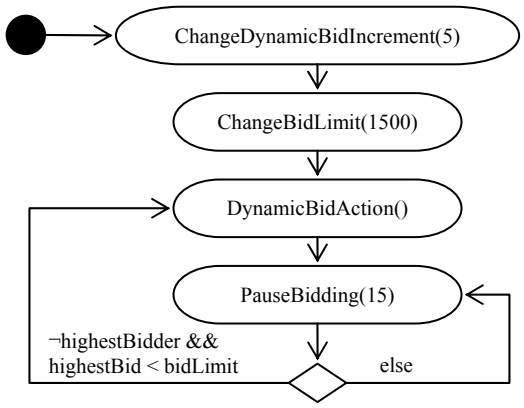

Figure 7. Simple strategy *S*2 (an aggressive one)

Figure 8 shows another simple strategy *S*3. According to this strategy, a bidding agent first changes the dynamic bid increment to $10. After placing the first dynamic bid, the bidding agent pauses for a random time from 2700 to 3600 seconds. It then checks to see if there is no new bid placed by another bidder during the past 900 seconds (i.e., 15 minutes). If so, the bidding agent places a dynamic bid; otherwise, it pauses for a random period of time again. This procedure continues until the bid limit is reached. Note that strategy *S*3 is a very cautious one, since it attempts

to avoid competing with other bidders and producing a high final bidding price.

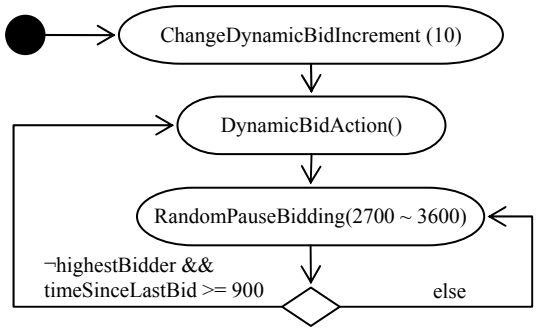

Figure 8. Simple strategy *S*3 (a cautious one)

In addition, we define two complex strategies *C*1 and *C*2. Complex strategy *C*1 is illustrated in Figure 5, which consists of simple strategy *S*1 and *S*2. Strategy *C*1 starts with aggressive simple strategy *S*1, but switches to an even more aggressive strategy *S*2 during the last 15 minutes of the auction. Complex strategy *C*2, illustrated in Figure 9, consists of simple strategy *S*1 and *S*3. Note that using *C*2, the bidding agent starts with aggressive simple strategy *S*1. Whenever there are more than five bidders actively placing bids in the auction, the bidding agent immediately switches to a cautious simple strategy *S*3. Thus we call complex strategy *C*2 an aggressive/cautious strategy.

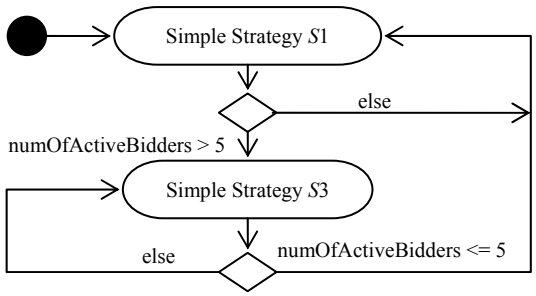

Figure 9. Complex strategy *C*2 (aggressive/cautious)

We ran simulations of agent-based online auctions using our specified bidding strategies. The simulations were designed to demonstrate how a user-specified bidding strategy may affect the behavior of a bidding agent and how it may directly impact the bid history of an auction. The platform we used is a prototype agent-based online auction system developed using JADE [15], where bidding agents can join and participate in multiple auctions at the same time according to user-specified bidding strategies. We run three auctions with the same fictitious auctioned items and the same auction duration of 48 hours. Each simulated auction contains six bidders: five bidders that are active through the whole auction, and one bidder that joins in the middle of the auction. For each of the three auctions, we vary the strategy used by bidding agent 1 as follows: For auction 1, bidding agent 1 utilizes complex strategy *C*1, for auction 2 it uses complex strategy *C*2, and for auction 3 it uses simple strategy *S*3. These three strategies are complex *aggressive* strategy, *aggressive/cautious* strategy, and *cautious* strategy, respectively. All the other five bidding agents in the three auctions use a *normal* strategy, which consistently outbids other bidders by a fixed increment (5 dollars) every 60 to 90 minutes using random pause bidding. Since all bidders except for bidder 1 use the same strategy, any major differences in the three auction's results should be primarily due to the different behaviors of bidding agent 1. Figure 10 demonstrates the bidding rate of agent 1 in the three auctions at each quarter of the auction durations.

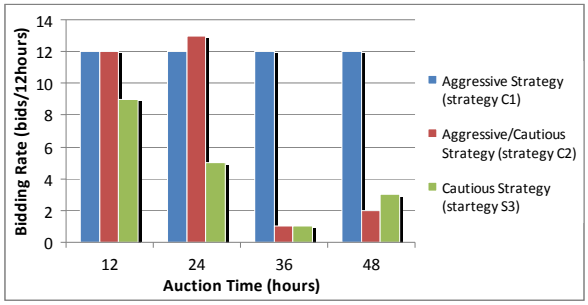

Figure 10. Bidding rates with different strategies

From the bar graph in Figure 10, we can see that the agent with an aggressive strategy has placed significantly more bids than the agent with a cautious strategy. However, for complex strategy *C*1, the transition from *S*1 to *S*2 at the last 15 minutes is not obvious. This is because the other five normal bidders typically pause for 60 to 90 minutes before they can place a new bid, and the agent using strategy *C*1 cannot outbid itself if it was the one to place the last bid. For cautious strategy *S*3, the agent has placed fewer bids during the third and fourth quarter than during the first and second quarter. This is due to the sixth agent joining in the middle of the auction, which makes overall bidding activities more frequent. For the aggressive/cautious strategy, we can clearly see that the strategy switches from an aggressive one to a cautious one after the sixth agent joins the auction, which causes the number of active bidders to reach six, and thus activates the strategy change in *C*2.

Figure 11 demonstrates the difference in bidding histories for the three auctions due to different strategies taken by bidding agent 1. When aggressive

strategy *C*1 is taken, the final bidding price is higher than the final prices in the other two auctions, in which *C*2 and *S*3 are taken, respectively. Note that the line graph shows the bidding activities of all six participating bidding agents. Conversely, when the cautious strategy *S*3 is taken by agent 1, the final bidding price is the lowest among the three auctions. Thus, it is easy to conclude that aggressive strategies may lead to higher final-bid prices.

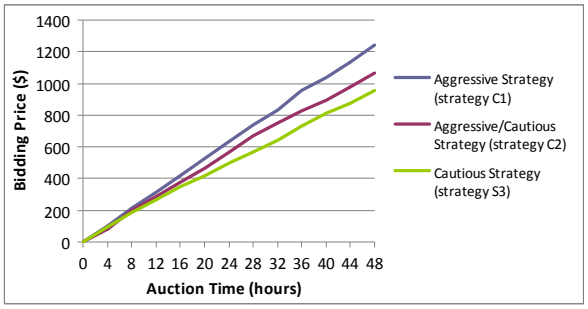

Figure 11. Bidding prices with different strategies

7. Conclusions and future work

In an agent-based online auction system, the efficient specification of bidding strategies is a necessary component to make the system practical and usable. In this paper, we present a model-based approach to specifying complex and flexible bidding strategies. Our layered structure of specified bidding strategies allows human users to easily mix and match various strategies to create their own complex ones. By converting the layered bidding strategy model into a rule-based bidding strategy model, the bidding strategy can be efficiently executed by the bidding agent. In addition, our approach may potentially support modification of bidding strategies by users at runtime, and real-time inclusion of bidding rules enforced by the auction house. In our future work, we plan to develop a graphical user interface (GUI) that supports visual specification of complex and flexible bidding strategies for bidding agents. The GUI will also allow users to modify bidding strategies that can be updated and executed by the bidding agent in real-time.

8. References

[1] P. Maes, R.H. Guttman, and A.G. Moukas, "Agents That Buy and Sell," *Communications of the ACM (CACM)*, Vol. 42, No. 3, March 1999, pp. 81-91.

[2] R. Patel, H. Xu, and A. Goel, "Real-Time Trust Management in Agent-Based Online Auction Systems," In *Proceedings of the Nineteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'07)*, Boston, USA, July 9-11, 2007, pp. 244-250.

[3] H. Xu, S.M. Shatz, and C.K. Bates, "A Framework for Agent-Based Trust Management in Online Auctions," In *Proceedings of the 5th International Conference on Information Technology: New Generations (ITNG 2008)*, April 7-9, 2008, Las Vegas, Nevada USA, pp. 149-155.

[4] J. Trevathan and W. Read, "A Simple Shill Bidding Agent," In *Proceedings of the 4th International Conference on Information Technology: New Generations (ITNG 2007)*, April 2-4, 2007, Las Vegas, Nevada, USA, pp. 766-771.

[5] S. Park, E.H. Durfee, and W.P. Birmingham, "An Adaptive Agent Bidding Strategy based on Stochastic Modeling," In *Proceedings of the Third Annual Conference on Autonomous Agents*, May 1999. Seattle, Washington, USA, pp. 147-153.

[6] M. He, N.R. Jennings, and A. Prügel-Bennett, "A Heuristic Bidding Strategy for Buying Multiple Goods in Multiple English Auctions," *ACM Transactions on Internet Technology (TOIT)*, Vol. 6, No. 4, November 2006, pp. 465-496.

[7] H. Ma and H. Leung, "An Adaptive Attitude Bidding Strategy for Agents in Continuous Double Auctions," *Electronic Commerce Research and Applications (ECRA)*, Vol. 6, No. 4, Winter 2007, pp. 383-398.

[8] M.P. Wellman, A. Greenwald, and P. Stone, *Autonomous Bidding Agents - Strategies and Lessons from the Trading Agent Competition*, The MIT Press, August 2007.

[9] T. Skylogiannis, G. Antoniou, N. Bassiliades, G. Governatori, and A. Bikakis, "DR-NEGOTIATE - A System for Automated Agent Negotiation with Defeasible Logic-based Strategies," *Data & Knowledge Engineering*, Vol. 63, No. 2, 2007, pp. 362-380.

[10] E. Gimenez-funes, L. Godo, J.A. Rodrguez-aguilar, and Pere Garcia-calves, "Designing Bidding Strategies for Trading Agents in Electronic Auctions," In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS 1998)*, July 1998, Paris, France, pp. 136-143.

[11] G. Governatori, M. Dumas, A.H. Ter Hofstede, and P. Oaks, "A Formal Approach to Protocols and Strategies for (Legal) Negotiation," In *Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL-2001),* May 2001, USA, pp. 168-177.

[12] E. Kontopoulos and N. Bassiliades, "Graphical Representation of Defeasible Logic Rules Using Digraphs," *Advances in Artificial Intelligence*, Vol. 3955, April 2006, pp. 529-533.

[13] M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, Third Edition, Addison-Wesley, 2004.

[14] J. Arlow and I. Neustadt, *UML and the Unified Process: Practical Object-Oriented Analysis and Design*, Addison-Wesley Professional, 2001.

[15] F.L. Bellifemine, G. Claire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*, John Wiley & Sons, Ltd., 2007.