On Market-Inspired Approaches to Propositional Satisfiability

William E. Walsh
University of Michigan AI Laboratory
1101 Beal Ave, Ann Arbor, MI
48109-2110, USA
wew@umich.edu

Makoto Yokoo
NTT Communication Science Laboratories
2-4 Hikaridai, Seika-cho, Soraku-gun,
Kyoto 619-0237, Japan
yokoo@cslab.kecl.ntt.co.jp

Katsutoshi Hirayama
Kobe University of Mercantile Marine
5-1-1 Fukaeminami-machi, Higashinada-ku,
Kobe 658-0022, Japan
hirayama@ti.kshosen.ac.jp

Michael P. Wellman
University of Michigan AI Laboratory
1101 Beal Ave, Ann Arbor, MI
48109-2110, USA
wellman@umich.edu

Abstract

We describe two market-inspired approaches to propositional satisfiability. Whereas a previous market-inspired approach exhibited extremely slow performance, we find that variations on the pricing method with a simplified market structure can improve performance significantly. We compare the performance of the new protocols with the previous market protocol and with the distributed breakout algorithm on benchmark 3-SAT problems. We identify a tradeoff between performance and economic realism in the new market protocols, and a tradeoff between performance and the degree of decentralization between the new market protocols and distributed breakout. We also conduct informal and experimental analyses to gain insight into the operation of price-guided search.

1 Introduction

Agents must often engage in activities with complex, interrelated dependencies. Even finding a satisficing solution for problems such as resource allocation, scheduling, and production in a supply chain is often intractable for a central problem solver with global knowledge. The problem is further complicated when decentralization constraints such as locality of interest, knowledge, communication, and authority must be respected.

Yokoo and Hirayama [2000], [Yokoo, 2000] formalize such problems as distributed constraint satisfaction problems (DisCSPs) and, with others, have designed a variety of effective algorithms. These approaches are generally distributed adaptations of centralized algorithms. Recent interest in market-based approaches to distributed decision making, and open questions about the computational power of markets [Shoham and Tennenholtz, to appear], prompted Walsh and Wellman [2000] to apply a market-based supply chain formation protocol [Walsh and Wellman, 1998] to a 3-SAT reduction of the supply chain formation problem, an approach they called MarketSAT. They found that, although market prices can guide decentralized search, the approach was impractically slow.

In this paper we present alternate, simpler market-inspired approaches that provide more satisfactory performance, while respecting well-defined decentralization constraints. We evaluate the protocols on benchmark propositional satisfiability (SAT) problems. As the fundamental NP-complete problem, formally equivalent to a large class of combinatorial problems, SAT serves as a convenient problem class on which to systematically evaluate our protocols.

In Section 2 we introduce two new MarketSAT protocols with qualitatively different pricing mechanisms. In Section 3 we provide an economic interpretation of the protocols and discuss the rationality of the assumed agent behavior. In Section 4 we convey our understanding of price-guided search and in Section 5 we show that the protocols are incomplete. In Section 6 we compare the performance of the new protocols with the original MarketSAT and with the distributed breakout (DB) algorithm [Yokoo and Hirayama, 1996]. We also perform further experiments to explain the operation of the protocols. In Section 7 we compare the decentralization of the MarketSAT protocols relative to distributed breakout. We conclude in Section 8.

2 MarketSAT Protocols

Following the notation of Walsh and Wellman [2000], we consider propositional satisfiability problems with variables U and clauses Q, each containing sets of literals over U in conjunctive normal form (CNF). A clause is satisfied if at least one literal in the clause evaluates to T. The problem is to determine whether there exists a truth assignment $t : U \to \{T, F\}$ that satisfies each $q \in Q$.

A variable u ***fails to satisfy*** a clause q under truth assignment t iff either: (1) $t(u) = T$, $u \notin q$, and $\bar{u} \in q$, *or*, (2) $t(u) = F$, $\bar{u} \notin q$, and $u \in q$. A MarketSAT economy consists of agents representing variable assignments and a set of goods specifying licenses to fail to satisfy clauses. Agents choose assignments for their corresponding variables, but must acquire the necessary licenses for their chosen truth assignments. Clearly, $q \in Q$ is satisfied under t iff *at most* $|q| - 1$ variables in q fail to satisfy q. Hence, we make available $|q| - 1$ such licenses.[1] It follows that a problem is satisfiable iff all agents can choose assignments such that they can obtain all necessary licenses to support their assignments.

Observe that our notion of failing to satisfy a clause is equivalent to the notion of a nogood in CSPs, which in SAT is just a negated clause. MarketSAT could be applied to more general CSPs by specifying the licenses to correspond to nogoods, with $|q| - 1$ licenses available for each nogood of size $|q|$.

In a MarketSAT protocol, the agents search for a satisfying solution in a decentralized fashion by negotiating to acquire the licenses. A market protocol consists of an auction mechanism and agent bidding policies. The negotiation for each license type is mediated by a separate auction. Although the two new MarketSAT protocols differ in the auction rules and bidding policies, they share common high-level features. A MarketSAT protocol iterates through the following steps:

1. Agents select tentative truth assignments and submit ***bids*** to a subset of the auctions corresponding to their chosen truth assignments.
2. Auctions send ***price quote*** messages to the agents indicating the current going prices of the licenses.

The protocol continues until ***quiescence***, a state in which no agent chooses to update any of its bids or the auctions terminate negotiation according to certain protocol-specific conditions. The auctions allocate their respective licenses to agents when quiescence is reached. For our empirical studies we assume synchronous instantaneous communication. Nothing in our protocols requires synchrony, but this allows us to fix certain parameters, focusing our studies.

The present market structure is a simplification of the original MarketSAT economy [Walsh and Wellman, 2000] based on a supply chain formation model [Walsh and Wellman, 1998]. In the original economy, there was a separate agent for positive and negative assignments of variables. An auction for each variable mediated the negotiation of the assignment agents to provide an assignment for the respective variables. An end consumer bid to acquire an assignment to each variable. Experiments show that the new protocols perform much faster with the new structure (Section 6).

In the following sections we describe the details of two new variants of MarketSAT.

2.1 Uniform Pricing

We describe a MarketSAT protocol with uniform pricing (MS-U) based on the original MarketSAT protocol (MS-O), but adapted to the present simplified structure. An auction allows an agent to place a bid to buy a license at a specified price. An agent may update its bid only if it increases the price of the bid by at least some publicly known increment δ. Agents may not withdraw bids.

Given a set of buy bids and $|q| - 1$ units of license g available (for the right to fail to satisfy a clause q), an auction reports a price quote comprised two parts: 1) the ***bid price***, $\beta(g)$, is the $(|q|)$th highest price of all bids, and 2) the ***ask price***, $\alpha(g)$, is the $(|q| - 1)$st highest price of all bids. If there are fewer than $|q| - 1$ bids in the auction, the bid and ask prices are zero. If there are $|q| - 1$ bids in the auction, the bid price is zero and the ask price equals the price of the lowest bid. The bid price specifies the price that the winning bidders would pay if the auction stopped in the current state and the ask price specifies how much a losing agent must bid in order to be a winning bidder. The price quote also indicates to a bidder whether its bid is one of the $|q| - 1$ highest (winning) bids. In quiescence, an auction allocates its units of license g according to the (M+1)st price auction rules [Wurman *et al.*, 1998]—the agents with the $|q| - 1$ highest bids win g and pay $\beta(g)$. For the price quotes and final allocation, the auction breaks ties in favor of earlier received bids, and randomly between simultaneously received bids.

An agent randomly chooses the initial assignment for its variable and places bids at price zero for the necessary licenses. When it subsequently receives price quotes, it chooses an assignment that minimizes its assignment cost, defined for an assignment as the maximum of the sum of its current ***perceived costs*** for the licenses needed for that assignment, and the previously computed assignment cost. In the case of ties, the agent keeps its current assignment. An agent's perceived cost for license g is $\beta(g)$ if it is winning, $\alpha(g)$ if it has not submitted a bid, $\alpha(g)$ if it is both losing and $\alpha(g) > \beta(g)$, and $\alpha(g) + \delta$ if it is both losing and $\alpha(g) = \beta(g)$.

After an agent chooses its assignment, it increments by δ any losing bid it has sent to the auction, if it needs the license for its chosen assignment. If it hasn't submitted a bid for some needed license, it submits a bid at price zero. An agent does not update its bids if it is winning all the licenses necessary for its current assignment. Observe that if the market

[1] In the original MarketSAT implementation [Walsh and Wellman, 2000], we used a reduction from 3-SAT and made 2 units of each license available. The existence of $|q| - 1$ licenses for a clause of size $|q|$ is a straightforward generalization.

reaches quiescence in this way, then the agents' local assignments constitute a globally satisfying assignment.

2.2 Differential Pricing

In this section we describe a MarketSAT protocol with differential pricing (MS-D). An auction allows an agent to place a bid to demand either either zero or one units of the license, without specifying a price. Agents may switch between zero and one quantity demands without constraint.

An auction may report different price quotes to each bidder, depending on the demand for the license. An auction maintains a nondecreasing ***premium price*** for its license. If the auction has $|q|-1$ units of the license available, and the total demand expressed by the bids is d, then the auction reports price quotes as follows:

- If $d < |q|-1$, then the auction reports a price of zero to all bidders.
- If $d = |q|-1$, then the auction reports a price of zero to all bidders with demand of one, and reports the premium price to the single bidder with demand of zero.
- If $d > |q|-1$, then the auction increases the premium by one, reports the premium price to one randomly chosen bidder with a demand of one, and reports a price of zero to all other bidders with demand of one.

In quiescence, if $d > |q|-1$, then the auction randomly allocates the license to $|q|-1$ agents that bid with demand one for the license, otherwise it allocates a license to each of the current bidders with demand one. Each agent that receives a license pays according the price specified in the last price quote it received.

An agent randomly chooses the initial assignment for its variable. When it subsequently receives price quotes, it chooses an assignment that minimizes the sum of the prices of licenses as reported by the last price quote, with preference for its current assignment when the costs are equal. When an agent is in its initial state, or when it flips its assignment, it places bids of quantity one for all licenses it needs for the assignment and places bids for quantity zero for all licenses necessary for the opposite assignment. When an agent does not flip its assignment, it resubmits any bid needed for that assignment and for which it received a premium price quote. If an agent does not flip its assignment and if received a zero price quote for all its bids for the assignment, the agent does not update any of its bids. Observe that if the market reaches quiescence in this way, then the agents' local assignments constitute a globally satisfying assignment. Furthermore, in this case every agent pays zero for the licenses it receives.

3 Economic Interpretation and Rational Behavior

The interpretation of the market-inspired protocols in economic terms requires a model of agent values under which the assumed behavior would be plausibly rational. For an agent a to be willing to participate in these protocols, and hence be willing to pay money for their allocations, it must obtain some individual value v_a from participating in a satisfying solution. An agent obtains no value if it does not participate in a satisfying solution. An agent wishes to maximize its surplus value, which is the difference between the value it obtains and the total price it pays for the licenses it acquires.

In both protocols, the bidding policies are non-strategic in that agents do not account for the behavior of other agents. This assumption may be reasonable in large networks for which agents have little information about other agent preferences and behavior.

The bidding policies are myopic and best-response in that agents always bid to optimize their surplus given the current price quotes, without speculating about future price changes. The plausibility of this approach depends on how accurately the price quotes indicate the prices agents may actually have to pay for their final allocations. For both protocols, the price quotes indicate what the agents would pay if the bidding stopped in the current state.

However, the protocols differ significantly in how price quotes signal future price movements. In MS-U, the nondecreasing price quotes indicate lower bounds on the amount that agents would have to pay, providing a basis for agents' belief in the price quotes. Since prices do not decrease, we would also expect that, in addition to the agent quiescence condition specified in Section 2.1, a rational agent would stop bidding when the total cost of an assignment exceeds v_a.

In MS-D, agents actually pay nothing for a globally satisfying allocation, and would have positive payments only if the protocol were to terminate with an unsatisfying allocation. Thus, unlike in MS-U, we would expect rational agents to stop bidding only when they reach a satisfying allocation (as specified in Section 2.2). But this calls into question the usefulness of the price quotes as meaningful future price indicators. The bidding policies in MS-D would be more plausibly rational if the agents had a reasonable expectation that the protocol could terminate at any time, for instance if the auctions terminated negotiations based on a random signal. Indeed, such a mechanism may be useful both to encourage desirable behavior and to bound the length of negotiations.

```
Initialize the weight of all nogoods to 1
UNTIL current state is solution DO
    IF current state is not a local minimum
    THEN make any local change that reduces
        the total weights of violated nogoods
    ELSE increase weights of all currently
        violated nogoods
END
```

Figure 1: The breakout algorithm [Morris, 1993].

4 Price-Guided Search

Intuitively, market prices indicate the relative global value of licenses, and agents use the prices to guide their local decisions. The bidding process can be seen as a distributed search for prices that support a satisfying allocation. In the MarketSAT protocols, prices are closely analogous to the weights of nogoods (which correspond exactly to failing to satisfy a clause) in the breakout algorithm [Morris, 1993], presented in Figure 1. In breakout, weights are a measure of how often the nogoods appear in local minima visited by the algorithm. The weights thus bias the search away from local minima.

In contrast to breakout, which increases the costs of nogoods only when the global state is a local minimum, the MarketSAT protocols increase the prices of licenses in response to local state. The price of a license increases whenever the corresponding clause is not satisfied for the current assignment, which can and does occur outside of local minima of the global state.

The breakout algorithm sequentially flips variable assignments to reduce the global weight of the violated nogoods. The nogood weights are counted only for clauses that are not satisfied. In MS-U, because bids cannot be withdrawn, once a license has an excess demand, it will always have excess demand, even if the current choices of variable assignments would indicate that the clause is currently satisfied. Thus the prices never decrease and agents attribute a cost to a license even if the clause is satisfied. In contrast, in MS-D an agent will only attribute a positive cost to a license if either the license is overdemanded or if flipping its own assignment would make the license overdemanded (assuming no other agent would also flip). In this sense, the cost evaluation in MS-D is closer to the breakout algorithm than in MS-U.

In breakout, current nogood costs are evaluated uniformly for all variables. In MS-U, an agent distinguishes its cost evaluation depending on whether it is winning or losing and assumes that its total cost over all licenses never decreases. This difference in perceived prices gives extra "friction" to the currently winning agents because they assume lower costs, hence are somewhat less likely to swap assignments. However, this friction is relatively small because agents increase their bids by δ only when they are losing, hence the bid and ask prices never differ by more than δ. In MS-D, agents can attribute widely varying costs to a license, with at most one agent attributing the premium cost and others attributing a zero cost.

Unlike breakout, variables can flip simultaneously in the MarketSAT protocols. This could be beneficial to performance when agents operate in parallel, if the right flips occur simultaneously. Yokoo and Hirayama [1996] observed that it could be detrimental if ***neighbor*** variables—those that share a clause—flip simultaneously, and thus incorporated synchronizing steps into DB to prevent simultaneous neighbor flips.

Of particular concern are ***simultaneous, satisfying neighbor flips***—neighbors simultaneously flipping to satisfy a clause (e.g., variables x and y simultaneously flipping to T to satisfy clause $(x \vee y)$) which we expect to be prevalent in the MarketSAT protocols. When neighbor variables simultaneously flip to satisfy the same clause, other clauses may become needlessly unsatisfied. Unlike DB, the MarketSAT protocols have no explicit mechanism to prevent simultaneous neighbor flips, yet we expect the different protocols to differ in the number of simultaneous, satisfying neighbor flips. In MS-U, because the bid/ask spread on any license is small, agents that desire common licenses are likely to have similar total cost evaluations. Hence, when the price of a shared license increases, they may be likely to simultaneously flip to satisfy the clause. In MS-D, because agents can attribute widely varying costs to licenses, the cost evaluations of neighbor agents are less likely to be close than in MS-U. We expect this would reduce the number of simultaneous, satisfying neighbor flips.

5 Completeness

We can show that MS-U is incomplete using a close variant of the example that Morris [1993] used to show that the (centralized) breakout algorithm is incomplete. We assume synchrony, noting that this implies the same for an asynchronous system. The CNF clauses are: $(\bar{x} \vee y)$, $(\bar{x} \vee z)$, $(\bar{x} \vee w)$, $(\bar{y} \vee x)$, $(\bar{y} \vee z)$, $(\bar{y} \vee w)$, $(\bar{z} \vee x)$, $(\bar{z} \vee y)$, $(\bar{z} \vee w)$, $(\bar{w} \vee x)$, $(\bar{w} \vee y)$, $(\bar{w} \vee z)$. Observe that the only solution for the problem assigns all variables T or all variables F. Consider the case when the initial random assignments are $t(x) = t(z) = T$ and $t(y) = t(w) = F$. Assume that the random tie breaking occurs as follows in the first round: x wins $(\bar{x} \vee y)$, y wins $(\bar{z} \vee y)$, z wins $(\bar{z} \vee w)$, and w wins $(\bar{x} \vee w)$. Assume that the random tie breaking occurs as follows in the second round: y wins $(\bar{y} \vee x)$, x wins $(\bar{w} \vee x)$, z wins $(\bar{y} \vee z)$, and w wins $(z \vee \bar{w})$. In all subsequent rounds, auction tie breaking is deterministic. Throughout, all agents flip simultaneously, hence the protocol oscillates indefinitely.

With the same SAT instance we used to show incompleteness of MS-U (but with different initial assignments and tie breakings), our simulation results strongly suggest to us that MS-O is not guaranteed to converge to a solution. We refrain from describing a trace of a non-converging run due to the greater complexity of the agent interactions in MS-O.

We can also show that MS-D is incomplete using Moris's exact example. The CNF clauses are: $(x \vee y \vee z \vee w)$, $(\bar{x} \vee y)$, $(\bar{x} \vee z)$, $(\bar{x} \vee w)$, $(\bar{y} \vee x)$, $(\bar{y} \vee z)$, $(\bar{y} \vee w)$, $(\bar{z} \vee x)$, $(\bar{z} \vee y)$, $(\bar{z} \vee w)$, $(\bar{w} \vee x)$, $(\bar{w} \vee y)$, $(\bar{w} \vee z)$. Observe that the only solution assigns all variables T. Consider the initial random assignment t, such that $t(x) = T$ and all other variables are F. The premium increases in each of the unsatisfied clauses so long as they remain unsatisfied, but the choice of which variable in the clauses pays the premium is chosen randomly. With positive probability, x can always be the only variable chosen to pay the premium, in which case it will flip indefinitely and no other variables will flip.

If the breakout algorithm reaches truth assignment t as described in the preceding, it *cannot* converge to the satisfying truth assignment [Morris, 1993]. But because MS-D has variable-specific pricing, it does not necessarily make a flip when it would be a global improvement, which can delay undesirable flips. In fact, MS-D can always converge, with positive probability, to some satisfying truth assignment t^*. With positive probability, an auction for a license corresponding to an unsatisfied clause will always report the premium price to some variable x such that $t(x) \neq t^*$. But then the protocol would clearly converge to t^*. We do not know if a similar result holds for MS-U.

6 Experiments

We compared the MarketSAT protocols with MS-O and DB using satisfiable (unforced, filtered) uniform random 3-SAT problems at the phase transition (between 4.26 and 4.3

clause/variable ratio) from the SATLIB benchmark library[2]. This is generally considered the hardest class of 3-SAT problems to solve [Cheeseman *et al.*, 1991; Mitchell *et al.*, 1992] and has been used widely to benchmark SAT algorithms. Note that, to generate hard satisfiable problems, it is important to generate problems randomly and filter the unsatisfiable instances. Forcing the problems to satisfy a particular assignment makes them much easier [Achlioptas *et al.*, 2000].

To bound the computational time, for a problem instance of n variables, we ran a protocol for at most $1000n$ rounds. For MarketSAT, a round is a synchronous bidding round, and for DB, a round corresponds to a cycle as described by Yokoo and Hirayama [1996]. Agents participate until they reach quiescence (with a satisfying solution), or until the maximum rounds is reached. We recognize that the artificial limit on the bidding rounds reduces the plausibility that agents may use the specified bidding policies (see Section 3), but chose this approach to consistently compare the performance of the protocols. We compared these protocols in terms of bidding rounds, which is the best measure of performance when agents operate in parallel. The performance is summarized in Table 1. Comparing the protocols from top to bottom, each subsequent protocol improves by about a factor of two to seven compared to the previous on all performance measures.

n	Success Ratio	Average Rounds	Median Rounds	σ Rounds
		Protocol: MS-O		
50	0.40	3.90×10^4	5.00×10^4	18.1×10^3
		Protocol: MS-U		
50	0.96	6.12×10^3	1.51×10^3	1.15×10^4
75	0.75	2.75×10^4	1.08×10^4	3.03×10^4
100	0.53	6.10×10^4	8.21×10^4	4.16×10^4
		Protocol: MS-D		
50	1.00	896	250	3.52×10^3
75	0.98	3.98×10^3	429	1.17×10^4
100	0.96	1.04×10^4	1.50×10^3	2.34×10^4
125	0.85	2.74×10^4	3.65×10^3	4.45×10^4
150	0.85	3.69×10^4	5.94×10^3	5.45×10^4
175	0.83	5.37×10^4	1.63×10^4	6.68×10^4
		Protocol: DB		
50	1.00	234	64.5	829
75	0.99	2.14×10^3	299	9.40×10^3
100	0.98	4.26×10^3	460	1.61×10^3
125	0.96	9.12×10^3	1.42×10^3	2.63×10^4
150	0.93	1.80×10^4	1.22×10^3	4.24×10^4
175	0.88	2.98×10^4	2.83×10^3	5.81×10^4

Table 1: Performance of protocols with n variables.

We attempted to determine the causes of differing performance of the protocols. We believe that the improvement of MS-U over MS-O is due to the simplified network structure, since MS-U is otherwise essentially the same as MS-O. One plausible conjecture for the difference between MS-U, MS-D, and DB is differing numbers of simultaneous, satisfying neighbor flips. We predict that an ordering of the protocols by increasing simultaneous, satisfying neighbor flips would be the same as by increasing performance.

To test this conjecture we measured the simultaneous, satisfying neighbor flips in both MS-D and MS-U. Also, recognizing that the MarketSAT protocols are much like breakout, but with simultaneous flips, we modified the breakout algorithm to allow simultaneous flips (with some ad hoc parameters to control the number of simultaneous flips and simultaneous, satisfying neighbor flips). To gain insight about the effects of simultaneous, satisfying neighbor flips, we tested breakout with simultaneous flips (SFB) with varying fractions of such neighbor flips. We also varied the number of generic simultaneous flips to help distinguish their contribution to performance from neighbor flips.

Protocol	Rounds	Flips	Flips Per Flip Round	Neighbor Flips Ratio
MS-U	6.12×10^3	1.02×10^4	2.21	0.33
MS-D	896	473	2.02	0.23
DB	234	218	1.41	0
SFB	483	329	1.00	0
	403	374	1.66	0
	494	390	1.42	0.02
	474	413	1.53	0.10
	483	432	1.65	0.13
	410	340	1.66	0.23
	408	468	1.99	0.37

Table 2: Comparison of simultaneous, satisfying flips for 50-variable problems.

Table 2 shows the average number of flips, flips per flip round (flips per round in which a flip actually occurred) and average fraction of simultaneous, satisfying neighbor flips for MS-U, MS-D, DB, and SFB. As expected, MS-U performs substantially more simultaneous, satisfying neighbor flips absolutely, and as a fraction of the total flips than does MS-D. However, the results from SFB do not support the conjecture that the neighbor flips contribute significantly to the performance difference. The performance of SFB does not appear to be sensitive to, or even monotonic in, the fraction of simultaneous, satisfying neighbor flips.

Table 2 suggests an alternate explanation of the performance difference between DB and MS-D. For 50-variable problems, MS-D requires nearly 3.8 times as many rounds as DB, but only 2.2 times as many flips. We also measured the number of rounds in which MS-D and DB performed flips and found found that, in fact, the average number of rounds in which variables actually flip in MS-D is 316, only 35% of the average total rounds. The average number of rounds in which DB performed a flip is 73% of its average total rounds. Thus it seems that the poor performance of MS-D relative to DB is due to the extra rounds to required to produce flips.

We conjecture that these extra rounds in MS-D happen be-

[2] `http://aida.intellektik.informatik.tu-darmstadt.de/SATLIB/benchm.html`

cause auctions randomly reassign the premium price at each round. Thus the agents' costs fluctuate randomly, and do not directly progress every time the premium increases. To verify this conjecture, we tested a variation of MS-D whereby pricing is reported as the premium and a fraction b of the premium (rather than the premium or zero as in MS-D). With a positive b, an agent's costs would not fluctuate so heavily from random assignments of the premium cost. We tried $b = 0.1$ on 50-variable problems and found that it required only 509 rounds, 462 flips, and 231 flip rounds. The ratio of rounds to flip rounds is only 2.2 with $b = 0.1$, compared to 2.8 for MS-D. Furthermore, although the variant with $b = 0.1$ outperformed MS-D, the fraction of satisfying neighbor flips was 0.43—significantly more than in MS-D. This evidence suggests that difference in performance between MS-D and DB is largely due to the extra rounds required to produce flips in MS-D, rather than simultaneous, satisfying neighbor flips.

Appealing to extra, non-flipping rounds does not seem to explain the relative performance difference between MS-U and MS-D, as the ratio of rounds to flip rounds is only 1.3 for MS-U, significantly less than in MS-D. An alternate explanation we propose is that the relatively poor performance of MS-U is due to the fact that costs continue to be attributed to a license when its respective clause becomes satisfied (note that prices are *increased* only for unsatisfied clauses though). If the prices do not distinguish between satisfied and unsatisfied clauses, it would seem that MS-U pricing may not provide a effective indication of the relative difficulty of satisfying a clause. In contrast, recall that, in MS-D, when a clause is satisfied, the agents currently demanding the associated license receive price quotes of zero. Breakout attributes no cost to satisfied clauses. To test whether attributing costs to satisfied clauses can be detrimental, we modified the breakout algorithm so that it does just that and found that the algorithm rarely found satisfying assignments. Of course MS-U did not perform this poorly, but it does differ from breakout in other ways, as described in Section 4. Still, our test suggests that we have identified a significant cause of the relatively poor performance of MS-U.

7 Discussion of Decentralization

The MarketSAT protocols are highly decentralized in the sense that agents need only know about and communicate with auctions for their own licenses (which in turn requires knowledge about the clauses in which they are contained) and auctions need only communicate with the agents that participate in them. Agents need not communicate with, or even know the existence of other variables. Similarly, auctions need not communicate with each other. In DB, an agent must know in which clauses its variable is contained and must also communicate with all variables in those clauses. MarketSAT can operate fully asynchronously. In DB, variables synchronize with their neighbors to detect quasi-local minima and to ensure that neighbor variables do not flip simultaneously.

In their previous work, Walsh and Wellman [2000] suggested that the highly decentralized nature of MS-O necessarily engenders poor performance. However, our experiments with MS-D suggest that a highly decentralized market-inspired protocol can perform reasonably well. Indeed, our experiments with variants on MS-D suggest that further improvements can be obtained with the same degree of decentralization. Still, it is an open question whether decentralized approaches could perform as well as the centralized SAT algorithms. Centralized algorithms can utilize techniques such as restarts (which contributed significantly to the success of GSAT [Selman *et al.*, 1992] and subsequent hill-climbing based algorithms) to help reduce heavy tails in the performance distribution. We found that restarts can also significantly improve the performance of MS-D. For example, with $10n$ max flips and 1000 max restarts, MS-D can solve all 175-variable problem instances within the bound, with average rounds 2.69×10^4, median 1.1×10^4, and $\sigma = 5.00 \times 10^4$. Although restarts could be implemented in a synchronous system with cooperating agents it is not obvious how such techniques might be utilized in a distributed, asynchronous system. Moreover, we do not have any intuition for an economic interpretation of restarts.

8 Conclusions

We described two market-inspired protocols for propositional satisfiability and compared them with the distributed breakout algorithm. We found that the pricing method can significantly affect the performance, with the differential pricing protocol about a half magnitude better than uniform pricing. However, the differential pricing protocol is less justifiable in terms of rational economic agent behavior. Although the MarketSAT protocols we consider perform significantly better than the original MarketSAT, the less decentralized distributed breakout algorithm still outperforms the differential pricing MarketSAT by a factor of three to four.

An informal analysis suggests that the price-guided search of MarketSAT works because the protocols resemble the centralized breakout algorithm. We found that the fraction of simultaneous, satisfying neighbor flips does not explain the difference in performance across protocols, but that the performance of MarketSAT with differential pricing suffers largely from the extra rounds needed to produce a flip.

We have identified tradeoffs in terms of runtime performance, decentralization, and the plausibility of assumed agent behaviors. Understanding these tradeoffs is necessary to make informed engineering decisions about the appropriateness and applicability of alternate decentralized approaches to a particular problem environment.

The market approach has the benefit of providing a price-based interface for an agent to evaluate and direct its behavior in the context of its broader decision making. To better understand and further develop market approaches to complex coordination problems, we must explicitly incorporate a model of the agents' economic motivations in the context of the problem to be solved. Future work should also include a deeper analysis of rational agent behavior.

Acknowledgments

The basic ideas for this project were developed during a visit to the University of Michigan by the second author. The authors wish to thank NTT and Edmund H. Durfee for supporting the visit. The first author was supported by a NASA/JPL Graduate Student Researcher fellowship.

References

[Achlioptas *et al.*, 2000] Dimitris Achlioptas, Carla Gomes, Henry Kautz, and Bart Selman. Generating satisfiable problem instances. In *Seventeenth National Converence on Artificial Intelligence*, pages 256–261, 2000.

[Cheeseman *et al.*, 1991] Peter Cheeseman, Bob Kanefsky, and William M. Taylor. Where the *Really* hard problems are. In *Twelfth International Joint Conference on Artificial Intelligence*, pages 331–337, 1991.

[Mitchell *et al.*, 1992] David Mitchell, Bart Selman, and Hector Levesque. Hard and easy distributions of SAT problems. In *Tenth National Conference on Artificial Intelligence*, pages 459–465, 1992.

[Morris, 1993] Paul Morris. The breakout method for escaping from local minima. In *Eleventh National Conference on Aritificial Intelligence*, pages 40–45, 1993.

[Selman *et al.*, 1992] Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. In *Tenth National Conference on Artificial Intelligence*, pages 440–446, 1992.

[Shoham and Tennenholtz, to appear] Yoav Shoham and Moshe Tennenholtz. On rational computability and communication complexity. *Games and Economic Behavior*, to appear.

[Walsh and Wellman, 1998] William E. Walsh and Michael P. Wellman. A market protocol for decentralized task allocation. In *Third International Conference on Multi-Agent Systems*, pages 325–332, 1998.

[Walsh and Wellman, 2000] William E. Walsh and Michael P. Wellman. MarketSAT: An extremely decentralized (but really slow) algorithm for propositional satisfiability. In *Seventeenth National Conference on Artificial Intelligence*, pages 303–309, 2000.

[Wurman *et al.*, 1998] Peter R. Wurman, William E. Walsh, and Michael P. Wellman. Flexible double auctions for electronic commerce: Theory and implementation. *Decision Support Systems*, 24:17–27, 1998.

[Yokoo and Hirayama, 1996] Makoto Yokoo and Katsutoshi Hirayama. Distributed breakout algorithm for solving distributed constraint satisfaction problems. In *Second International Conference on Multi-Agent Systems*, pages 401–408, 1996.

[Yokoo and Hirayama, 2000] Makoto Yokoo and Katsutoshi Hirayama. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems*, 3(2):198–212, 2000.

[Yokoo, 2000] Makoto Yokoo. *Distributed Constraint Satisfaction: Foundation of Cooperation in Multi-agent Systems*. Springer, 2000.