ICGA Journal 40 (2018) 406–416
DOI 10.3233/ICG-180063
IOS Press

Mixed logical and probabilistic reasoning in the game of Clue

Todd W. Neller * and Ziqian Luo
Department of Computer Science, Gettysburg College, PA, USA

Abstract. We describe a means of mixed logical and probabilistic reasoning with knowledge in the popular game Clue. Using pseudo-Boolean constraints we call *at-least constraints*, we more efficiently represent cardinality constraints on Clue card deal knowledge, perform more general constraint satisfaction in order to determine places where cards provably are or are not, and then employ a WalkSAT-based solution sampling algorithm with a tabu search metaheuristic in order to estimate the probabilities of unknown card places. Finding a tradeoff between WalkSAT-heuristic efficiency in finding solution samples and the sampling bias such a heuristic introduces, we empirically study algorithmic variations in order to learn how such sampling error may be reduced.

Keywords: Clue, Cluedo, at-least constraints, cardinality constraints, extended clauses, sampling, logical reasoning, probabilistic reasoning, WalkSAT, tabu search

1. INTRODUCTION

Clue®[1] is a mystery-themed game of deduction (Fig. 1). The goal of the game is to be the first player to correctly name the contents of a case file: the murder suspect, the weapon used, and the room the murder took place in. There are 6 possible suspects, 6 possible weapons, and 9 possible rooms, each of which are pictured on a card. One card of each type is chosen randomly and placed in a "case file" envelope without being revealed to any player. All other cards are dealt out face-down to the players. Each player takes on the identity of one of the suspects.

Each player thus begins with private knowledge that their dealt cards are not in the case file. As the game proceeds, players suggest possible suspect, weapon, and room combinations, and other players refute these suggestions by privately revealing such cards to the suggester. This type of game is called a *knowledge game* van Ditmarsch (2000), and the basic knowledge of the game may be expressed using propositional logic.[2]

However, while basic knowledge of the game is expressible using propositional logic, there are particular game facts that are not efficiently expressible in propositional logic. In this paper, we will describe a generalization of CNF SAT knowledge representation making use of a specialization of pseudo-Boolean constraints that have been called *cardinality constraints* Dixon et al. (2004), but we will call *at-least constraints* in order to avoid confusion with other uses of that term. After briefly sketching the basic game knowledge of Clue in Section 2 and the constraint reasoning used to determine where cards provably are or are not in Section 3, we largely focus on our generalized SAT sampling used to estimate the probabilities of uncertain card-place pairs. In Section 4, we generalize

*Corresponding author. E-mail: tneller@gettysburg.edu.

[1] A.k.a. Cluedo® in other parts of the world.

[2] Van Ditmarsch has examined reasoning of Clue using dynamic epistemic logic (van Ditmarsch (2002); van Ditmarsch and Kooi (2015)), but we will show that even the probabilistic estimation of card positions using propositional knowledge presents practical computational difficulties.

Fig. 1. The game of Clue.

SAT sampling, measure sampling bias introduced by efficient sampling (4.1), and perform an empirical comparative study of sampling algorithm variations in seeking to reduce sampling error (4.2). Conclusions and future work are discussed in Section 5.

2. BASIC KNOWLEDGE OF CLUE

In this section we describe the deduction game of Clue and characterize the basic propositional knowledge used to reason about the game.

The murder-mystery themed board game of Clue was invented by Anthony E. Pratt and first produced in the U.K. by Waddingtons as "Cluedo" in 1949. Since that time, Clue became one of the most popular family board games of the last half century.

In the back-story of the game, Mr. Boddy has been murdered in one of the rooms of his mansion Tudor Close. The game's object is to be the first to deduce the correct murder suspect, weapon, and room. Three to six players arbitrarily assume roles of possible suspects, but a player's guilt or innocence does not affect play.

A deck of 21 cards depicts each of six possible suspects, six possible weapons, and nine possible rooms. One of each of these three card types is randomly chosen and placed, unseen by any player, into an envelope called the *case file*. These three cards are the unknown murder suspect, weapon, and room for this game. The remaining 18 cards are then shuffled and dealt clockwise to players. (In a four- or five-player game, cards will not be dealt evenly.)

Each turn, a player rolls dice to move about the board, visiting rooms and making suggestions about the contents of the case file. When a player makes a suggestion, opponents clockwise indicate that they cannot disprove the suggestion until an opponent has one or more of the suggested cards and must reveal one privately to the suggester. Each player may declare one accusation in the game, checking

the case file to see if the contents have been correctly named. If correct, the player wins; if not, the player loses and continues to disprove suggestions when possible.

Thus, Clue is primarily a game of information, and successful players are skilled at gaining more information than their opponents, using it to deduce or guess further information. Atomic sentences for our propositional reasoning are of the form "Card c is in place p," denoted c_p.

Students often share fond memories of playing Clue as children. Indeed, Clue is often considered a simple children's game. This is perhaps due to the fact that the game's detective notepads are too small to make substantial notes, and instructions for use of detective notepads merely encourage the player to simply mark the cards the player has seen. However, most information available in the game concerns cards *not* seen. It is important to note what is *not* held by an opponent, or that one of three suggested cards was shown by one opponent to another. In our experiences, few Clue players make such notes or appreciate the logic puzzles each game presents.

The knowledge of Clue can be divided into two types: initial knowledge and play knowledge. Initial knowledge is that which is known right before the first turn and includes:

- Each card is in exactly one place.
- Exactly one card of each category is in the case file.
- You know your hand of cards.
- You know how many cards have been dealt to each player.

Play knowledge is knowledge gained during events of the game:

- A player cannot refute a suggestion.
- A player refutes your suggestion by showing you a card.
- A player refutes another player's suggestion by showing them a card privately.
- A player makes an accusation and shares whether or not the accusation was correct.

3. LOGICAL REASONING WITH AT-LEAST CONSTRAINTS

Whereas almost all basic knowledge of Clue may be easily expressed in propositional logic, e.g. DIMACS format CNF, There is one type of information above that enlarges the representation size combinatorially: "You know exactly how many cards have been dealt to each player." Consider a 6-player game where each player has exactly 3 of the 21 cards. To express the fact that *at most* 3 cards are held by a player p, for every set of 4 different cards $\{c_i, c_j, c_k, c_l\}$, one knows $\neg c_{i\,p} \vee \neg c_{j\,p} \vee \neg c_{k\,p} \vee \neg c_{l\,p}$. Given 6 players and 21 cards, this amounts to $6 \times \binom{21}{4} = 35{,}910$ such clauses added to our knowledge base.

To express the fact that *at least* 3 cards are held by a player p, for every set of 19 different cards, one can form a disjunction of c_p for each c of those 19 cards, knowing that at least 1 of every 19 cards must be be held by p. Given 6 players and 21 cards, this amounts to $6 \times \binom{21}{19} = 1{,}260$ such clauses added to our knowledge base.

There are, of course, alternate ways of encoding such cardinality constraints, yet each has its own combinatorial complexity. In our algorithmic approach, we choose instead to generalize SAT CNF clauses to more naturally and compactly express such constraints. Consider the SAT clause, i.e. disjunction, $(l_1 \vee l_2 \vee \cdots \vee l_n)$, often denoted $\{l_1, l_2, \ldots, l_n\}$ for brevity, where each l_i is a literal, i.e. an atomic sentence or its negation. At least one of the literals must be true.

We generalize this to what we will call an *at-least constraint*, denoted $\{l_1, l_2, \ldots, l_n\}_m$, where at least m of the n literals must be true. Then, the expression of the fact that exactly 3 cards are held by player can be expressed by two at-least clauses, the first expressing that at least 3 of the 21 literals c_p for each card c must be true, and the second expressing that at least 18 of the 21 literals $\neg c_p$ must be true. Thus, we only need 12 at-least clauses to express the knowledge that each of 6 players have exactly 3 cards.

One may accordingly generalize DPLL-style SAT solvers in order to do such reasoning, but we have extended Donald Knuth's Dancing Links (DLX) algorithm Knuth (2000) for more general constraint satisfaction with at-least constraints.

4. PROBABILISTIC ESTIMATION THROUGH SAMPLING

$P(c_p)$ is the fraction of models (i.e. card deals consistent with the game knowledge) in which c_p is true. However, for much of a 6-player game, the combinatorial number of models makes such model counting prohibitively expensive. Instead, for each card whose place is unknown, we *estimate* the probability of the card being at a place by searching for and collecting card deals consistent with the at-least constraints of our knowledge base, and calculating the fraction of such deals where the card is at that place. Put another way, we search for and sample models of our knowledge base, and estimate probabilities of card-place atomic sentences according to the frequency of such atomic sentences being true among the models found.

Throughout the remainder of the paper, we will refer to such a model (i.e. consistent/satisfying deal) as a *sample solution* or *sample*.

Our basic WalkSAT-like algorithm is described in Algorithm 1. Note that "clauses" in this context are at-least clauses rather than simple disjunctions. In its most basic form, it operates similarly to the WalkSAT algorithm Selman et al. (1994), but with two main differences. First, search continues for a fixed number of iterations, not terminating with the first sample found, but rather continuing to find more samples within the allotted budget of iterations. Second, we make use of a metaheuristic idea from Tabu Search Glover (1989, 1990). After a flip in variable assignment, the variable may not, in general, be reflipped until after a fixed number of steps we call the *tabuTenure*, so as to avoid immediate duplicate samples. That is, a variable may not generally be reflipped until *tabuTenure* iterations have passed.

Even with such a tabu mechanism discouraging immediate repeat samples, we can find some samples visited repeatedly while others are never found. It is important to observe that *the efficiency of finding samples via a WalkSAT-style search comes as a cost of sampling bias.* It is with this tension between *sampling efficiency* and *bias minimization* that we concern ourselves as we empirically evaluate the following variations to our basic algorithm.

4.1. Measurement of sampling error

In order to compare the error reduction of various search-and-sample algorithms, we must first establish our error measure.

We randomly generated 100 complete simulated games of Clue. For each game, we took the game information from the winner's perspective and reduced suggestion information to a minimal subset necessary in order to deduce the contents of the case file. Then, we iteratively eliminated suggestion

Algorithm 1 Search and sample – a WalkSAT-like sampling algorithm

Input: at-least clauses of the game state, fixed variable assignments, number of search iterations
Output: probability estimate of each card being at each place

function SearchAndSample(*clauses*, *fixedVarAssignments*, *numIter*):
RandomRestart(); {randomly assign all non-fixed variables}
if there are no unsatisfied clauses **then**
  RecordSample(); {tally true non-fixed variables, increment sample count}
end if
tabuTenure $= 10$
lastSampleStep $= 0$
for *step* $= \{1, 2, 3, \ldots,$ *numIter*$\}$ **do**
  tabuCutoffStep $= \max(1,$ *step* $-$ *tabuTenure*, *lastSampleStep*$)$
  if all clauses are satisfied **then**
    Choose a random non-fixed variable.
  else
    Choose a random unsatisfied clause.
    Choose a variable from that clause according to the WalkSAT heuristic.
  end if
  Flip(); {make chosen variable flip assignment, update data structures}
  if there are no unsatisfied clauses **then**
    RecordSample();
    lastSampleStep $=$ *step*
  end if
end for

information from the end of the game backwards, thus increasing the existing number of samples. For each elimination iteration, we applied our generalization of Knuth's Dancing Links (DLX) algorithm to search out all samples and compute the exact probability of each card-place pair. If, while searching, the number of samples was found to exceed a threshold of 10,000,000, we terminated iteration and continued with the next simulated game.

For each game state where we are able to compute exact probabilities for card-place pairs, we first mark all such atomic-sentence variables as *fixed* that have probabilities of 0 or 1. Then we call our search-and-sample algorithm for 100,000 iterations in order to estimate probabilities for all non-fixed variables.

Our uncertainty measure is then the root-mean-square deviation (RMSD) of all non-fixed variable estimates compared to their exact probabilities. This search-and-sample algorithm and RMSD calculation is performed 100 times, and we compute the mean RMSD for each game state.

Thus, for the 539 later-game states of the 100 Clue games we simulated, we executed our search-and-sample algorithm to collect samples and estimate card-place probabilities 100 times, and then reported the mean RMSD as our measure of sampling error.

4.2. Algorithmic variations

In this section, we briefly describe our algorithm variations, present data, and then for each describe the motivation for the variation, summary error results, and our interpretation thereof.

(a) Random Restart After finding and recording a sample solution, perform a random restart, reinitializing all non-fixed variable to random values.

(b) Random Flip or Restart After finding and recording a sample solution, perform a random variable flip with probability 0.2. otherwise, perform a random restart.

(c) Mixed Random/Heuristic Flip Selection After having chosen a random unsatisfied clause, with probability 0.2, flip a random variable of that clause. Otherwise, flip a random variable among those that minimize the number of clauses that will become false as a result.

(d) Eliminate Duplicate Solutions Record only unique sample solutions.

(e) Reduce Tabu Tenure In addition to unique samples, reduce the *tabuTenure* constant from 10 to 2, allowing greater frequency of individual variable flips.

4.2.1. Random restart

On the iteration after finding a sample, Algorithm 1 will flip a random variable, remaining local to that sample and may find itself trapped in a local minimum. As a middle ground between generate-and-test and our WalkSAT-like tabu search, we attempted a random restart after finding each sample (Fig. 2(a)). Whereas our base algorithm's average RMSD was 0.01021 our variation with random restarts after each sample had a greater average RMSD of 0.01316 We observed that this variation wastes too many iterations traversing variable assignments to reach local neighborhoods of solutions, only to reset to yet another long traversal.

4.2.2. Random flip or restart

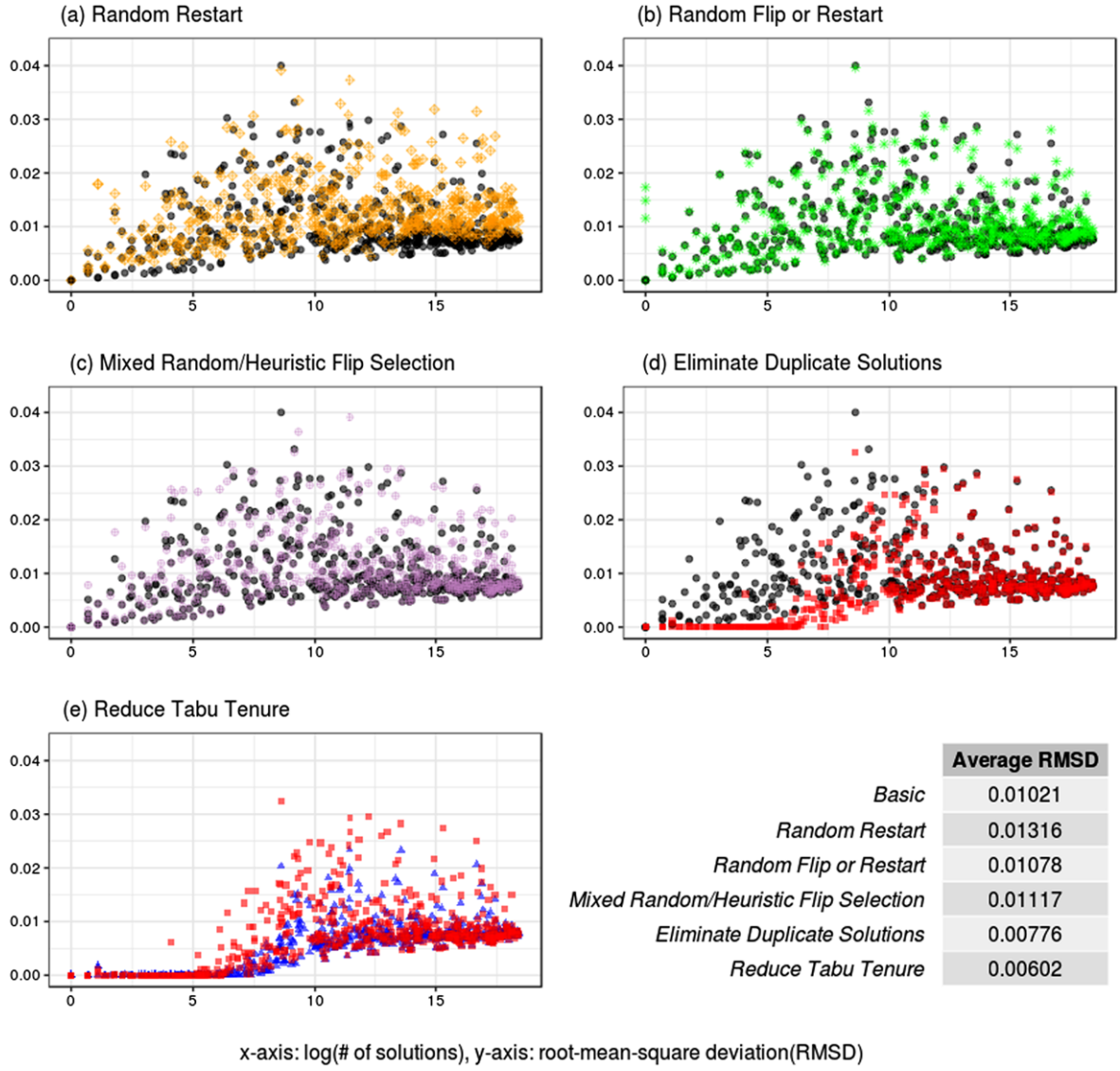
As a moderation of the previous idea, we next tried to have this random restart occur only 20% of the time a sample is recorded, and retaining basic algorithm behavior otherwise (Fig. 2(b)). Compared to the basic algorithm mean RMSD of 0.01021 this variation has an increased RMSD of 0.01078. Having tried random restarts with different frequencies and only observing increased error, we thus abandoned random restarts as a means of diversifying our sample.

4.2.3. Mixed random/heuristic flip selection

Our next experimental variation was to introduce noise into our selection of variable flips in order to see if this improved our sample distribution. With probability 0.2, we perform a random flip of a non-fixed variable from a random unsatisfied clause, rather than randomly choosing among those variables that, after flipping, would cause the minimum number of clauses to become unsatisfied (Fig. 2(c)). Again, we observed an increase of RMSD from 0.01021 for our basic algorithm to 0.01117 for this variation. We observe that, while the efficiency of our WalkSAT-like search step in finding samples leads to sampling bias, the aforementioned introductions of randomness in search detrimentally decreases the number of samples found within our limited budget of search iterations and thus increases error due to decreased sampling.

4.2.4. Eliminate duplicate solutions

We next experimented with the discarding of duplicate samples, leaving all other aspects of the algorithm the same, including tabu search behavior as if the sample was recorded (Fig. 2(d)). Here we observe our first improvement. From the basic algorithm mean RMSD of 0.01021 eliminating duplicated samples reduced mean RMSD by 24% to 0.00776. While this in no way increases the diversity of samples found, this nonetheless avoids over-representation of more easily found samples.



	Average RMSD
Basic	0.01021
Random Restart	0.01316
Random Flip or Restart	0.01078
Mixed Random/Heuristic Flip Selection	0.01117
Eliminate Duplicate Solutions	0.00776
Reduce Tabu Tenure	0.00602

Fig. 2. Error results from all approaches. In subfigures (a)–(d), dark/light points are RMSD values of the basic/variant algorithm, respectively. In subfigure (e), algorithms (d) and (e) are compared with light/dark points, respectively. regression curves fitting these points are shown in Fig. 3.

4.2.5. Reduce tabu tenure

For our final algorithmic variation, we similarly eliminated duplicate samples and also experimented with various *tabuTenure* values in {1, 2, 3, 4, 8, 10}, finding that our least error occurred with *tabuTenure* = 2 (Fig. 2(e)). This further reduces mean RMSD from the previous value of 0.00776 by 22.4% to 0.00602 It should be noted that, lacking a tabu search mechanism altogether, the algorithm functions very poorly, becoming trapped into duplicate resampling. However, we learned that only a very little tabu tenure is necessary to reap its greatest benefits in this context.

All algorithm RMSD curves fitted with LOESS regression[3] (and limited to non-negative values) are given in Fig. 3.

[3]From R's ggplot2 package.

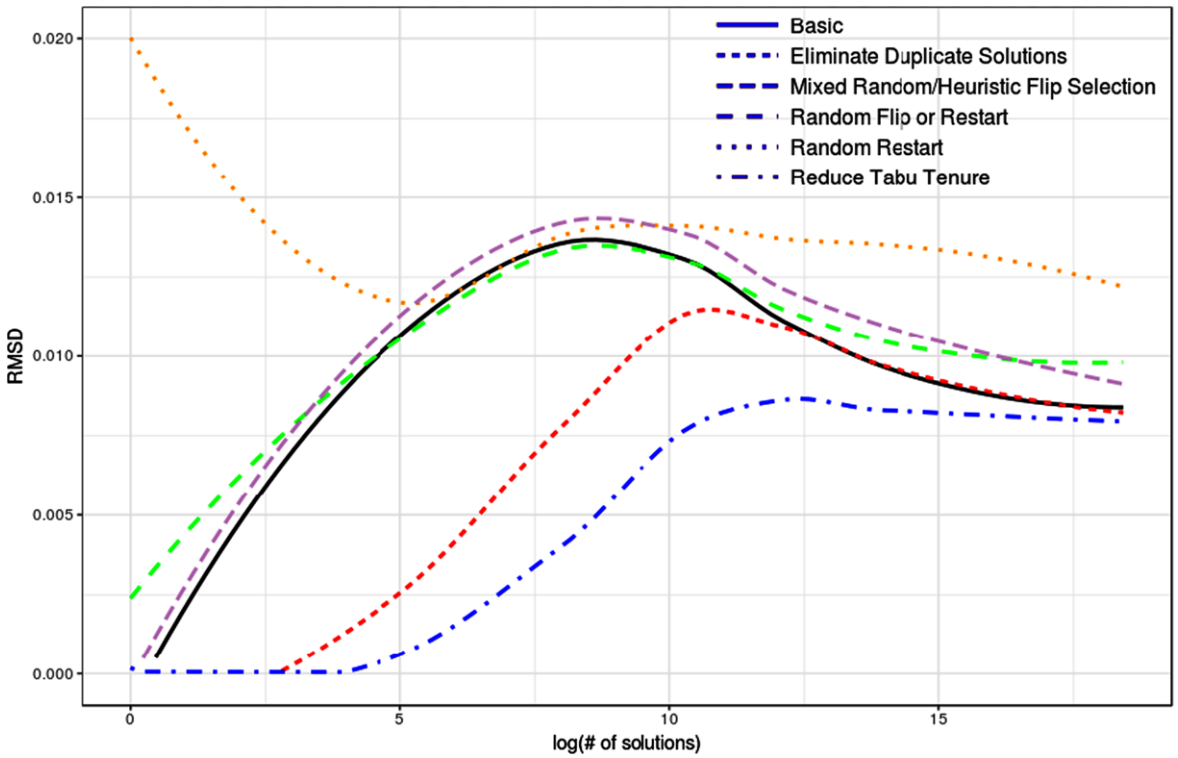


Fig. 3. RMSD regression curves for all search-and-sample algorithms.

4.3. Algorithm 2: Best search and sample

The best of these search-and-sample algorithms we will call Algorithm 2. Algorithm 2 is modified from Algorithm 1 as follows:

Algorithm 2 Best search and sample (changes listed from Algorithm 1)

- Line 9: Replace with $tabuTenure = 2$
- Line 21: Replace with:

  if this sample has not already been recorded
  | *RecordSample();*
  end if

To better understand the practical effect of these changes, we present in Table 1 an example comparison of probabilistic exact values and estimates from Algorithms 1 and 2. Table 1(a) provides abbreviated game information from the perspective of the Colonel Mustard (mu) player. Players are seated in the order given by the columns of Tables 1(b)–(d), so the first line of Table 1(a) indicates that Mustard (mu) suggested that the murder was committed by suspect Scarlet (sc) with the revolver (re) in the ballroom (ba), but that this suggestion was refuted by player Scarlet (sc), who showed the ballroom card (ba underlined). Since Scarlet is just before Mustard in the turn order, this implies that all other players could not refute the suggestion and thus had none of the three suggested cards.

With all information of Table 1(a) there are still 11 cards whose locations are uncertain. The exact probabilities are given to two decimal places in Table 1(b), with probabilities 1 and 0 represented by “O” and “X”, respectively. (These certainties are computed by our at-least constraint reasoning.)

Table 1
Example estimation of card location probabilities

Suggester	Refuter	Suspect	Weapon	Room
mu	sc	sc	re	ba
wh	pe	wh	ca	li
gr	pe	gr	re	ha
sc	wh	pe	ca	lo
wh	gr	wh	pi	di
wh	sc	pe	ro	lo
gr	pe	pl	ca	bi
sc	pl	pe	re	st
mu	gr	gr	ro	bi
wh	mu	sc	re	li

(a) Example game play information

	sc	mu	wh	gr	pe	pl	cf
mu	0.13	X	0.31	0.18	0.11	0.27	X
pl	X	O	X	X	X	X	X
gr	X	X	X	O	X	X	X
pe	O	X	X	X	X	X	X
sc	X	X	X	X	X	X	O
wh	**0.13**	X	**0.30**	X	**0.30**	**0.26**	X
kn	0.13	X	0.31	0.18	0.11	0.27	X
ca	0.02	X	0.11	X	0.83	0.04	X
re	X	X	X	X	X	X	O
ro	X	O	X	X	X	X	X
pi	0.06	X	0.14	0.61	0.05	0.13	X
wr	0.13	X	0.31	0.18	0.11	0.27	X
ha	X	X	X	X	O	X	X
lo	0.02	X	0.93	X	X	X	0.05
di	0.06	X	0.14	0.47	0.05	0.13	0.14
ki	0.10	X	0.22	0.13	0.08	0.20	0.26
ba	O	X	X	X	X	X	X
co	0.10	X	0.22	0.13	0.08	0.20	0.26
bi	0.10	X	X	0.13	**0.27**	0.20	0.29
li	X	O	X	X	X	X	X
st	X	X	X	X	X	O	X

(b) Exact probabilities

	sc	mu	wh	gr	pe	pl	cf
mu	0.90	X	0.33	0.18	0.90	0.31	X
pl	X	O	X	X	X	X	X
gr	X	X	X	O	X	X	X
pe	O	X	X	X	X	X	X
sc	X	X	X	X	X	X	O
wh	**0.20**	X	**0.24**	X	**0.36**	**0.20**	X
kn	0.16	X	0.30	0.17	0.10	0.28	X
ca	0.02	X	0.17	X	0.76	0.04	X
re	X	X	X	X	X	X	O
ro	X	O	X	X	X	X	X
pi	0.12	X	0.16	0.53	0.06	0.14	X
wr	0.09	X	0.33	0.18	0.09	0.31	X
ha	X	X	X	X	O	X	X
lo	0.05	X	0.88	X	X	X	0.07
di	0.04	X	0.13	0.55	0.04	0.11	0.13
ki	0.06	X	0.24	0.13	0.07	0.21	0.29
ba	O	X	X	X	X	X	X
co	0.10	X	0.23	0.13	0.07	0.21	0.26
bi	0.07	X	X	0.14	**0.37**	0.18	0.25
li	X	O	X	X	X	X	X
st	X	X	X	X	X	O	X

(c) Probabilities from Algorithm 1

	sc	mu	wh	gr	pe	pl	cf
mu	0.14	X	0.31	0.17	0.10	0.28	X
pl	X	O	X	X	X	X	X
gr	X	X	X	O	X	X	X
pe	O	X	X	X	X	X	X
sc	X	X	X	X	X	X	O
wh	**0.13**	X	**0.29**	X	**0.33**	**0.25**	X
kn	0.13	X	0.31	0.18	0.10	0.28	X
ca	0.02	X	0.12	X	0.81	0.04	X
re	X	X	X	X	X	X	O
ro	X	O	X	X	X	X	X
pi	0.06	X	0.15	0.62	0.05	0.13	X
wr	0.14	X	0.31	0.18	0.10	0.28	X
ha	X	X	X	X	O	X	X
lo	0.02	X	0.92	X	X	X	0.05
di	0.07	X	0.15	0.46	0.05	0.13	0.14
ki	0.09	X	0.22	0.14	0.08	0.20	0.27
ba	O	X	X	X	X	X	X
co	0.10	X	0.22	0.13	0.08	0.21	0.27
bi	0.10	X	X	0.13	**0.30**	0.20	0.27
li	X	O	X	X	X	X	X
st	X	X	X	X	X	O	X

(d) Probabilities from Algorithm 2

Probabilities estimated by Algorithms 1 and 2 are given in Tables 1(c) and 1(d), respectively. Let us highlight a few observations from these estimates.

Note first how much improved the estimates are for the probable locations of the White (wh) card. Boldface estimates in the "wh" row of Table 1(c) have errors that are both significant and repeatable across runs of Algorithm 1. Comparing this with the Algorithm 2 estimates, we can see that our estimates with duplicate sample elimination and reduced tabu tenure are much closer to exact values.

Note also that the greatest errors in our Algorithm 2 estimates as boldfaced in the Peacock (pe) columns of Tables 1(b)–(d) are roughly half the magnitude of the worst errors observed with Algorithm 1, and that such errors are much less common.

5. CONCLUSIONS

In this empirical study, we described a means of mixed logical and probabilistic reasoning with knowledge of the popular game Clue. Using pseudo-Boolean constraints we call *at-least constraints*, we generalize the common CNF SAT representation for propositional logic in order to more efficiently represent cardinality constraints on card deal knowledge. Logical reasoning to determine known variable values is accomplished through a constraint satisfaction generalization of Knuth's Dancing Links algorithm. Following this, we estimate the probabilities of unknown card places through a WalkSAT-based solution sampling algorithm with a tabu search metaheuristic.

However, the efficiency of finding and sampling solutions is also the cause of sampling bias. We then conducted an empirical study of algorithmic variations in order to reduce sampling error. Two ideas, incorporated in Algorithm 2, resulted in a total 41% reduction of root-mean-square deviation in estimation error: (1) elimination of duplicate samples, and (2) reduction in tabu tenure. As we learned, the tabu metaheuristic was indeed important to the successful operation of the algorithm, and yet the best tabu tenure was a short tenure for this problem domain. Further, we learned that seeking a more diverse sample through the introduction of various forms of randomness came at an even greater cost of error through much-reduced sampling.

5.1. Future research

We have shown uncertain reasoning in the game of Clue to be interesting in a few respects. First, cardinality constraints on card hands leads to a combinatorially large representation in SAT, and thus challenges us to consider algorithms for reasoning more directly with generalizations of SAT constraints. Second, except for end game states, game states generally admit a combinatorially large number of satisfying models (i.e. card deals) such that exact model counting is computationally expensive, suggesting a need for probabilistic estimation through sampling. Third, the very WalkSAT-like heuristic that helps us find such samples quickly also unfortunately introduces a sampling bias. Whereas such sampling bias is observed in similar work on the SAT problem (Gomes et al. (2009)), it appears to be a more serious problem for application in this problem domain.

This work represents initial steps to mitigate such sampling bias and compute better probabilistic estimates efficiently. However, we would expect that future work could improve upon this work in two important respects described in Gomes et al. (2009): (1) estimation quality, and (2) confidence bounds on such estimations. Through tuning of tabu tenure and eliminating duplicate samples, we have made improvements in the first respect, and we expect further improvements are possible. To

our knowledge, there is no prior work on computing confidence bounds on such at-least constraint sampling estimates.

Such confidence bounds are of interest in assessing the utility of making, for example, an uncertain accusation when one believes one may not get another turn to make a certain accusation in Clue. More generally, probabilistic estimates in constraint satisfaction reasoning will become more useful for decision-making when we can better characterize expected error in such estimates.

REFERENCES

Dixon, H.E., Ginsberg, M.L. & Parke, A.J. (2004). Generalizing Boolean satisfiability I: Background and survey of existing work. *Journal of Artificial Intelligence Research*, *21*, 193–243. https://www.aaai.org/Papers/JAIR/Vol21/JAIR-2107.pdf. doi:10.1613/jair.1353.

Glover, F. (1989). Tabu search – Part I. *ORSA Journal on Computing*, *1*(3), 190–206. https://doi.org/10.1287/ijoc.1.3.190. doi:10.1287/ijoc.1.3.190.

Glover, F. (1990). Tabu search – Part II. *ORSA Journal on Computing*, *2*(1), 4–32. https://doi.org/10.1287/ijoc.2.1.4. doi:10.1287/ijoc.2.1.4.

Gomes, C.P., Sabharwal, A. & Selman, B. (2009). Model counting. In A. Biere, M. Heule, H. van Maaren and T. Walsh (Eds.), *Handbook of Satisfiability* (pp. 633–654). IOS Press. Chap. 20, http://www.cs.cornell.edu/~sabhar/chapters/ModelCounting-SAT-Handbook-prelim.pdf. doi:10.3233/978-1-58603-929-5-633.

Knuth, D.E. (2000). Dancing links. In J. Davies, B. Roscoe and J. Woodcock (Eds.), *Millenial Perspectives in Computer Science*. Cornerstones of Computing (pp. 187–214). Macmillan Education UK. https://arxiv.org/abs/cs/0011047. ISBN 9780333922309.

Selman, B., Kautz, H.A. & Cohen, B. (1994). Noise strategies for improving local search. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)* (Vol. 1, pp. 337–343). Menlo Park, CA, USA: AAAI Press.

van Ditmarsch, H. & Kooi, B. (2015). *One Hundred Prisoners and a Light Bulb* (pp. 109–121). Springer. Chap. 11.

van Ditmarsch, H.P. (2000). *Knowledge Games*. PhD thesis, Groningen University, Groningen, Netherlands. ILLC Dissertation Series 2000-06.

van Ditmarsch, H.P. (2002). The description of game actions in Cluedo. In L.A. Petrosian and V.V. Mazalov (Eds.), *Game Theory and Applications* (Vol. 8, pp. 1–28). Nova Science Publishers.