ON THE NP-HARDNESS OF BLOCKS WORLD

Stephen V. Chenoweth

NCR Research & Development
1700 S. Patterson Blvd.
Dayton, Ohio 45479

Abstract

Blocks world (cube world) has been one of the most popular model domains in artificial intelligence search and planning. The operation and effectiveness of alternative heuristic strategies, both basic and complex, can be observed easily in this domain. We show that finding an optimal solution is NP-hard in an important variant of the domain, and popular extensions. This enlarges the range of model domains whose complexity has been explored mathematically, and it demonstrates that the complexity of search in blocks world is on the same level as for sliding block problems, the traveling salesperson problem, bin-packing problems, and the like. These results also support the practice of using blocks world as a tutorial search domain in courses on artificial intelligence, to reveal both the value and limitations of heuristic search when seeking optimal solutions.

Introduction

Blocks world is a model domain used in artificial intelligence to explore different approaches to automated reasoning--especially heuristic search and planning. Examples of such use begin with Winograd's (1971) "micro-world" for the simulated robot SHRDLU, and have continued to appear in the literature ever since.

Usually blocks world exemplifies that a given algorithm can perform planning, or that it is efficient in terms of the number of calculations required to find a solution or in terms of the length of that solution. Nilsson (1980), for instance, used blocks world in these ways to describe the advantages and disadvantages of deductive approaches, production systems, the STRIPS planner, and other alternatives.

Despite its flair for portraying the effects of different strategies, blocks world has not been used in the same way as sliding block (8-Puzzle, etc.) or other model domains for studies of complexity. In large part, this may be attributed to the fact that the inherent difficulty of the domain was unknown. The current study resolves this issue for an important variant of the domain and several extensions: The optimization problem for these is NP-hard.

Typical Domain Description in the Literature

Winograd's "micro-world" and Fahlman's (1974) similar robot planning domain included blocks of various shapes and sizes, strewn on a flat surface, occasionally on each other, sometimes colored, but always having perfectly flat surfaces. Nilsson (1980) and most other authors have since emphasized a very simple variant, also called cube world, in which only cube-shaped blocks of identical sizes are moved by a single robot arm. We consider here this version of the domain.

Research work using this simple cube domain includes Sacerdoti (1975), Sussman (1975), Soloway and Riseman (1977), Chapman (1985, 1987), Ratner and Warmuth (1986), Rolston (1987), Ginsberg (1988), Hogge (1988), Wilkins (1988), Kambhampati (1990a, 1990b), Zlotkin and Rosenschein (1990), Drummond (1990) and McDermott (1990).

The domain also serves tutorial purposes in artificial intelligence courses, and is used to introduce planning in many AI texts. In addition to Nilsson (1980), examples include Charniak, *et al* (1980), Winston and Horn (1981), Rich (1983), O'Shea and Eisenstadt (1984), Haugeland (1985), Charniak and McDermott (1985), Luger and Stubblefield (1989), and Schalkoff (1990).

In the basic version of this domain, cube-shaped blocks of equal size are moved by a mechanical arm from a "start state" to a predefined "goal state." The arm can move a single block at a time. Stacks are formed by placing a block precisely on top of another block.

A "table" (or "floor") is provided where blocks may be placed, to start new stacks or as a temporary resting place to give access to others stacked beneath them. Blocks can be moved *from* either the table or the top of a stack, *to* either the table or the top of a stack. For simplicity, we view a "move" from one state to another in the domain as an entire completed action of this type (versus, say, having a move be just picking up a block). We count each such move as having unit cost.

The objective of a blocks world problem is to create a "route" describing how to move the blocks, one at a time, from start state to goal state. As the blocks are moved, a series of intermediate states are formed. Some states may be required to reach the goal state, even though they involve moving a block to something other than its desired position in the goal state.

An example of a simple blocks world problem is depicted in Figure 1 (omitting the arm, and the block shapes themselves). In the start state, block A is on top of block B, which is on the table, and block C is on the table. In the goal state, block C is on block B, which is on block A, and block A is on the table.

```
                      C
  A                   B
  B     C             A
-------------     -----------
 Start State       Goal State
```

Figure 1. A simple blocks world problem.

A possible route for solving this problem would be to have the mechanical arm pick up A and set it on the table, then pick up B and set it onto A, and finally pick up C and set it onto B.

Most authors begin with a domain like that just described, to show the effects of planning algorithms and heuristics. Some go on to extend the domain to something like the micro-world of early authors, adding different shapes of blocks, multiple arms, and the like. Treatment of the domain has been in the language of each authors' knowledge representation methods.

Examples in the literature tend to deal with fairly simple planning problems such as that pictured in Figure 1. The simplicity of this figure, though, is quite misleading. In problems involving many blocks, the optimal route usually is well hidden, and methods of finding such a route without search inevitably seem to work only on problems up to a certain size. Similar problems often have quite different solutions.

For example, in Figure 2, the thirteen-move solution that begins by unstacking K, A, and B onto the table, from the "already correct" stack K, A, B, C, turns out to be one move shorter than any route that leaves this stack alone. Yet, in the similar problem shown in Figure 3, the 10-move solution that reassembles blocks D, E, F, G, and H is one move shorter than any route that unstacks blocks from K, A, B, C.

```
K  D              K  F
A  E  G  I        A  E  H  J
B  F  H  J        B  D  G  I
C  A  B  C        C  A  B  C
-------------    --------------
 Start State       Goal State
```

Figure 2. A more difficult blocks world problem.[1]

[1]The duplication of block names in each state is intentional.

```
K  D           K  F
A  E  G        A  E  H
B  F  H        B  D  G
C  A  B        C  A  B
-----------   -----------
Start State   Goal State
```

Figure 3. A problem similar to Figure 2, but with a very different optimal route.

Current Study

Like in most model AI domains, every problem in this basic version of blocks world has a trivial solution, so long as its quality is not an issue[2]. All blocks not already correctly positioned for the goal state simply could be set off onto the table (one at a time with the mechanical arm), and then reassembled in the proper order on top of any blocks already correctly positioned. As for these other common domains, the blocks world optimization problem appears to be much harder. Unlike them, however, a demonstration of complexity for this problem has not been given in the literature.

In the current study we show that this appearance of complexity is no illusion: In at least one important variant of the domain and many of its extensions, the optimization problem is NP-hard. Sections 2 and 3 of the paper lead to the main conclusion, while Section 4 discusses the extensions and Section 5 concludes.

Assumptions and Definitions

Three additional assumptions are given, which were not spelled out in the basic version of blocks world above:

1. Multiple blocks of the same type are allowed (as shown in Figures 2 and 3). This is a point on which authors using the domain typically have been non-committal. Those using this feature to bring out specific points about planning include Wilkins (1988), and Zlotkin and Rosenschein (1990). We call such blocks "interchangeable."
2. Only a single mechanical arm is allowed. The majority of work with blocks world has assumed this.
3. The table is assumed to have infinite capacity. This is consistent with its usage in the literature.

We formulate a blocks world problem, **B**, by defining the stacks of blocks which comprise the start state and goal state. Intermediate states are similarly defined for a given succession of moves in a route. Block positions in these states can be described as ordered pairs: (A, B) means block A is on block B. Moves are described by defining a block in the start or intermediate state at hand, and the new destination for that block.

[2]E.g., this is true of traveling salesperson or sliding block problems.

Theorem

We show that finding an optimal solution is NP-hard in the basic version of blocks world characterized by the three assumptions of Section 2, by proving the following equivalent statement:

Theorem. The question, "Given a blocks world problem, **B**, is there a route of *M* moves?" is NP-complete in the strong sense.

Proof. Given a supposed route for **B**, clearly we can decide in polynomial time if it is legal and contains *M* moves. Thus, the question is in NP.

We show that an arbitrary instance of the domain 3SAT can be transformed into an instance of the blocks world question in a polynomial number of steps[3]. Let *C* = {*c1, c2,* ... , *cm*}, *U* = {*u1, u2,* ... , *un*} define an arbitrary instance of 3SAT. Then we define the following corresponding blocks world problem (ref. Figure 4):

1. The start state includes blocks having these positions (where each letter/subscript combination represents a unique type of block), arranged into stacks as shown:

a. For each *ui* in *U*:

- (vi, table)
- (ui, zri1), (zri1, zri2), ... , (zril, wi), (wi, table), where zri1, ..., zril represent types of blocks for indexes r_{i1}, ..., r_{il} of clauses containing the literal "*ui*".
- (ui, zsi1), (zsi1, zsi2), ... , (zsil, xi), (xi, table), where zsi1, ..., zsil represent types of blocks for indexes s_{i1}, ..., s_{il} of clauses containing the literal "not *ui*".[4]

b. For each *cj* in *C*:

- (cj, sj), (sj, table)

c. The stack (R, v1), (v1, v2), ..., (vn, V)

d. (S, table)

2. The goal state includes blocks having these positions, arranged into stacks as shown:

1. For each *ui* in *U*:
   - (ui, vi), (vi, table)
   - (ui, vi), (vi, table) (i.e., a duplicate)
   - (wi, table); and (xi, table)
2. For each *cj* in *C*:
   - (cj, zj), (zj, table)
   - (zj, zj), (zj, table)
3. The stack (R, s1), (s1, s2), ..., (sm, S)
4. (V, table)

[3]3-Satisfiability (Cook, 1971), a restricted version of the Satisfiability problem which is often used for NP-completeness proofs, is defined as follows (Garey and Johnson, 1979, p. 46):

Instance: Collection *C* = {*c1, c2,* ... , *cm*}, of disjunctive clauses on finite set *U* = {*u1, u2,* ... , *un*} of variables such that |*ci*| = 3 for all *i*.

Question: Is there a truth assignment for *U* that satisfies all the clauses of *C*?

[4]Giving 3 z's of any one type, total--one for each variable in a clause.

Each of these states contains $6n+5m+3$ blocks, with the same numbers of each type. Thus, the construction defines a polynomial transformation from a 3SAT instance to a blocks world instance. This also is a polynomial time transformation, whose time is at most a function of $3mn + 6n + 2m + 3$ (a linear function of the number of blocks in the problem, except for the z's, which require analyzing at most all *m* clauses for each of the *n*-*u*'s).

What remains to be shown is that the question for 3SAT also transforms into a question for the blocks world problem, for a suitable value of *M*, such that the answer for one is "yes" if and only if the answer for the other is "yes."

For *M* we pick the value $3n+5m+1$, which is the number of blocks in the problem that cannot be in the correct position, no matter how a route is selected. (This includes R, the *n*-v's stacked on V, 2*n*-u's, *m*-s's, *m*-c's, and 3*m*-z's.) Clearly, all these must be moved during a route, so $3n+5m+1$ is the minimum number of moves that any route could have. Furthermore, a route of such length can be created only if no other moves are required in the route; in particular, no moves can be made in such a route unless they place a block in its final destination in the goal state. We call these "moves to goal." Our plan is to show that the problem can be solved using only moves to goal if and only if there is a truth assignment for *U* that satisfies all the clauses in *C*.

Suppose that a satisfying truth assignment exists, giving literal values *t1, t2,* ..., *tn* to *u1, u2,* ..., *un,* respectively. Then for each value *ti* of *ui* we proceed as follows:

1. For each successive *ui* in *U*, we do these moves:

- If *ti* = true, then we place onto vi the ui which is on top of the z's, created for "ui". Then we place each of these z's onto the table if there are not already two matching z's on the table; otherwise we place the z on top of one of the matching z's.
- If *ti* = false, then we place onto vi the ui which is on top of the z's, created for "not ui." We also place all the z's on the table or on a matching z, as described above.

2. For each successive *cj,* we then move cj onto a zj. We know that this can be done for all *cj*'s, since the *ui*'s selected satisfy all the 3SAT clauses: There must be at least one top block zj on the table for each cj to be moved to.

3. We then stack all the sj's, in the required order, onto block S.

4. We move R from v1 onto s1.

5. We move all the vi's that are in block V's stack onto the table.

6. For each *i*, we move the remaining ui onto a vi, and all the remaining z's below it onto the table (if no matching top-block zi is on the table), or else onto a matching z block (if such a top-block exists).

```
                 u1       u1   u2   u2  ...  un   un    R
                 z's      z's  z's  z's ...  z's  z's   v1
                 for u1   for   .    .        .    .    v2
                 clauses  not   .    .        .    .    .
                 .        u1    .    .        .    .    .
                 .        .                             .
                 .        .                             vn       c1  c2 ... cm
v1 v2 ... vn     w1       x1   w2   x2  ...  wn   xn    V    S   s1  s2 ... sm
```

Start State

```
                                                                   R
                                                                   s1
                                                                   s2
                                                                   .
                                                                   .
                                                                   .
u1  u1  ...  un  un                          c1 ... cm   z1 ... zm   sm
v1  v1  ...  vn  vn   w1 ... wn   x1 ... xn  z1 ... zm   z1 ... zm   S     V
```

Goal State

Figure 4. The constructed problem for the theorem.[5]

This solves blocks world problem **B** in the required $3n+5m+1$ moves.

Now suppose that no truth assignment for U will satisfy the clauses of C. We still can move one or the other ui block onto a vi (as pictured in the start state in Figure 4), in order to free z's for placing cj's, etc. However, no combination of picking these ui's can result in allowing all cj blocks to be moved to a goal position. Since none of the top blocks in the problem other than the ui's can be moved without a move that is not to a goal position, there is no route for this problem that can avoid a required non-goal move (such as moving a ui to table). We conclude that the problem cannot be solved with a route of M moves.

Since the size of the maximum number used in an instance of this question is less than the number of symbols used, clearly the problem is NP-complete in the strong sense. **Q.E.D.**

From the proof of the theorem, we can derive that it remains true even if there are no more than 3 blocks of any one interchangeable type (the z's in the proof).

[5]The relative physical position of stacks on the table is disregarded in these two state pictures.

Extensions and Reductions

These results clearly extend to a variety of more complicated blocks worlds, so long as these include the one described in the proof as a subset. Examples are the following:

1. Allowing different block shapes and sizes (Fahlman, 1974).
2. Removing the table as a place for temporary block storage, making the satisficing problem more difficult (Davis, 1986). (The use of the table in the theorem proof can be replaced by placing them on blocks created for that purpose.)
3. Providing either multiple possible goal states or multiple possible start states for a problem.
4. Using multiple arms.
5. Allowing each block to have multiple, alternative roles (perhaps implemented by allowing the arm to rotate blocks so that different faces show).

However, if P $\neq$ NP, such extensions guarantee that problems of non-polynomial time complexity will be encountered only if care is taken to include all problems of the configuration required for the proof of NP-completeness in the smaller domain. Otherwise, a separate proof of complexity is required. For example, tutorial problems in which the table was restricted might naturally exclude significant problems in the domain allowing any use of the table.

In the same way, blocks world problems that represent or extend restrictions of a larger domain may or may not be NP-hard. For example, Chapman (1985) showed that "the problem of determining whether a proposition is necessarily true in a nonlinear plan whose action representation is sufficiently strong to represent conditional actions, dependency of effects on input situations, or derived side effects is NP-hard." Chapman's proof (p. 45) relies on showing that any propositional formula p on atoms p_i can be represented as a plan in the domain. For a specific planning domain, such as blocks world, this may not be the case.

Conclusions

This paper extends the range of AI model domains whose complexity has been explored mathematically. It demonstrates that the complexity of search in blocks world is on the same level as for sliding block problems (Ratner and Warmuth, 1986), and classic NP-hard problems like the traveling salesperson problem and bin packing (Garey and Johnson, 1979). This allows the domain to be selected, for demonstrating search and planning method efficiency with large optimization problems.

These results also support the practice of using blocks world as a tutorial search domain in texts and courses on artificial intelligence. For such usage, the clarity with which effectiveness of alternative strategies can be seen in this domain is now enhanced by its known complexity.

The question of NP-hardness remains open in domain versions for which the one used here is *not* a subset. Worst case complexity results have (so far) not been demonstated for the optimization problem if the first of the Section 2 assumptions is omitted, for instance. Chenoweth (1986) proposed a general problem-solving method yielding "good" solutions for this version, with worst case N^2 complexity, where N is the number of blocks in a problem. A variation of this algorithm including search for an optimal solution was shown to run in polynomial time if the number of stacks allowed in the start state was fixed.

The formal study of complexity of approximation algorithms remains to be done. The simplest of these, unstacking and then restacking all the incorrectly positioned blocks, yields a solution quality no worse than twice optimal. Slightly more sophisticated, perhaps, is progressively to move blocks to their goal position when such opportunities present themselves; else move arbitrary incorrectly positioned blocks to the table. The average case solution quality of these approximation algorithms has yet to be investigated.

Acknowledgements

Thanks to Prof. Terry McKee of Wright State University for his great assistance in developing these ideas.

References

Chapman, D., 1985. *Planning for Conjunctive Goals*. Report No. 802, MIT. Excerpts also under same title in *Artificial Intelligence* 32, 1987, pp. 333-377.

Charniak, E., *et al.*, 1980. *Artificial Intelligence Programming*. Hillsdale, New Jersey: Lawrence Erlbaum Associates, pp. 258-282.

Charniak, E., and McDermott, D., 1985. *Introduction to Artificial Intelligence*. Reading, Mass.: Addison-Wesley Publ. Co.

Chenoweth, S.V., 1986. Mathematical Foundations for Blocks World Including a Proof of NP-Completeness. Masters Thesis, Department of Computer Science and Engineering, Wright State University.

Cook, S.A., 1971. The Complexity of Theorem-Proving Procedures. *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, Association for Computing Machinery, New York, pp. 151-158.

Davis, H.W., 1986. Private conversation, Wright State University.

Drummond, M.E., 1990. Refining and Extending the Procedural Net. *Readings in Planning*, ed. by J. Allen, *et al.* San Mateo, California: Morgan Kaufmann Publishers, Inc., pp. 667-669.

Fahlman, S.E., 1974. A Planning System for Robot Construction Tasks. *Artificial Intelligence* 5(1), pp. 1-49.

Garey, M.R., and Johnson, D.S., 1979.*Computers and Intractability, a Guide to the Theory of NP-Completeness*. New York: W.H. Freeman and Company.

Ginsberg, M.L., and Smith, D.E., 1988. Reasoning About Action II: The Quantification Problem. *Artificial Intelligence* 35, pp. 311-342.

Haugeland, J., 1985. *Artificial Intelligence: The Very Idea*. Cambridge, Mass.: MIT Press.

Hogge, J.C., 1988. Prevention Techniques for a Temporal Planner. *7th AAAI*, pp. 43-48.

Kambhampati, S., 1990a. Mapping and Retrieval During Plan Reuse: A Validation Structure Based Approach. *9th AAAI*, pp. 170-175.

Kambhampati, S., 1990b. A Theory of Plan Modification. *9th AAAI*, pp. 176-182.

Luger, G.F., and Stubblefield, W.A., 1989. *Artificial Intelligence and the Design of Expert Systems*. Reading, Mass.: Addison-Wesley Publishing Co.

McDermott, D., 1990. Planning and Acting. *Readings in Planning*, ed. by J. Allen, *et al.* San Mateo, California: Morgan Kaufmann Publishers, Inc., pp. 225-244.

Nilsson, N., 1980. *Principles of Artificial Intelligence.* Palo Alto, California, Tioga Press.

O'Shea, T., and Eisenstadt, M., ed., 1984. *Artificial Intelligence: Tools, Techniques, and Applications.* Cambridge: Harper & Row.

Ratner, D. and Warmuth, M., 1986. Finding the Shortest Solution for the NxN Extension of the 15-Puzzle is Intractable. *Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 168-172.

Rich, E., 1983. *Artificial Intelligence.* New York: McGraw-Hill Book Co.

Rolston, D.W., 1987. Toward a Tensed-Logic-Based Mitigation of the Frame Problem. *The Frame Problem in Artificial Intelligence: Proceedings of the 1987 Workshop*, pp. 319-342.

Sacerdoti, E.D., 1975. *A Structure for Plans and Behavior.* Menlo Park, California: SRI AI Center Technical Note 109.

Schalkoff, R.J., 1990. *Artificial Intelligence: An Engineering Approach.* New York: McGraw-Hill Publishing Co.

Soloway, E.M., and Riseman, E.M., 1977. Levels of Pattern Description in Learning. *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, pp. 801-811.

Sussman, G.J., 1975. *A Computer Model of Skill Acquisition.* New York: American Elsevier.

Winograd, T., 1971. *Procedures as a Representation for Data in a Computer Program for Understanding Natural Language.* MIT: Project MAC Technical Report No. 84 (Ph.D. Dissertation). Revised version later published as *Understanding Natural Language*; New York: Academic Press, 1972.

Wilkins, D.E., 1988. *Practical Planning: Extending the Classical AI Planning Paradigm.* San Mateo, Cal.: Morgan Kaufmann Publ., Inc.

Winston, P.H., and Horn, B.K.P., 1981. *Lisp.* Reading, Mass.: Addison-Wesley Publishing Co., pp. 179-218.

Zlotkin, G., and Rosenschein, J.S., 1990. Negotiation and Conflict Resolution in Non-Cooperative Domains. *8th AAAI*, pp. 100-105.