

# Merge-and-Restart Meta-Agent Conflict-Based Search for Multi-agent Path Finding

David Tolpin,  
dtolpin@robots.ox.ac.uk

## Abstract

We introduce a new algorithm for multi-agent path finding, derived from the idea of meta-agent conflict-based search (MA-CBS). MA-CBS is a recently proposed algorithm for the multi-agent path finding problem. The algorithm is an extension of Conflict-Based Search (CBS), which automatically merges conflicting agents into meta-agents if the number of conflicts exceeds a certain threshold. However, the decision to merge agents is made according to an empirically chosen fixed threshold on the number of conflicts. The best threshold depends both on the domain and on the number of agents, and the nature of the dependence is not clearly understood.

We suggest a justification for the use of a fixed threshold on the number of conflicts based on the analysis of a model problem. Following the suggested justification, we introduce a new algorithm, which differs in the ways when and how meta-agents are created and handled during search. The new algorithm exhibits considerably better performance compared to the original algorithm. The new algorithm is evaluated on several sets of problems, chosen to highlight different aspects of the algorithm.

## Introduction

In the Multi-Agent Path Finding (MAPF) problem, we are given a graph  $G(V, E)$  and a set of  $N$  agents  $a_1 \dots a_N$ . Each agent  $a_i$  has a start position  $s_i \in V$  and a goal position  $g_i \in V$ . At each time step an agent can either move to a neighboring location or wait in its current location, at some cost. The objective is to return a least-cost set of actions for all agents, which will move all of the agents from start to goal positions goal without conflicts (i.e., without any pair of agents *being in the same node* or *crossing the same edge* at the same time). MAPF has practical applications in robotics, video games, aviation, vehicle routing, and other domains (Silver 2005; Wang, Botea, and Kilby 2011). In its general form, MAPF is NP-complete, since it is a generalization of the sliding tile puzzle, an NP-complete problem, but a solution can still be verified in a polynomial number of steps (Ratner and Warmuth 1986).

In this paper we consider a particular variant of MAPF, for which Meta-Agent Conflict-Based Search (MA-CBS) (Sharon et al. 2012b), the algorithm explored here,

was formulated. The total solution cost is the sum of costs of all actions (and hence the sum of costs of solutions for each of the agents). Any single action, as well as waiting during a single time step in a non-goal position, has unit cost. Waiting in the goal position has zero cost. The problem is solved in *the centralized computing* setting, where a single program controls all of the agents<sup>1</sup>.

MA-CBS is a generalization of Conflict-Based Search (CBS) (Sharon et al. 2012a). MA-CBS presents a compromise between CBS and completely coupled solvers, such as A\*, A\*+OD (Standley 2010), or EPEA\* (Felner et al. 2012). The first step of MA-CBS is identical to that of CBS, with low-level search performed by a single-agent search algorithm. At every following search step MA-CBS employs a *heuristic*: if the number of conflicts for a pair of agents exceeds a certain threshold  $B$ , MA-CBS merges the two agents into a combined agent. Experimental results showed that for certain values of the threshold MA-CBS outperforms both CBS and single-agent search. However, threshold  $B$  used in the heuristic has to be empirically determined, and varies both with the size and shape of graph  $G$  and with the number of agents  $N$ . Difficulty choosing the ‘right’ value for  $B$  limits practical usability of MA-CBS.

Generally, a heuristic represents abstraction or approximation of a phenomenon associated with the problem or algorithm. Understanding why a particular heuristic works helps make better decisions involving the heuristic. One way to discover powerful heuristics for a particular problem is to design them systematically (Prieditis 1993; Hernavolyi and Holte 2004). However, a heuristic can also come as an insight, and in this case explaining why the heuristic is successful helps further improve the algorithm.

In this paper we look at the heuristic decision-making of MA-CBS, in which a fixed threshold on the number of conflicts between a pair of agents is used to replace the agents with a single combined agent. Based on the observations of the dependence of the threshold on features of the problem, we suggest an explanation for the threshold, and propose a model problem where the decision can be made optimal in a certain sense of optimality. Based on the model problem,

---

<sup>1</sup>This setting is tantamount to decentralized cooperative setting with full knowledge sharing and free communication (Sharon et al. 2012b).

we empirically investigate variants of MA-CBS. The investigation

- provides further support for the hypothesis regarding the root cause behind the fixed threshold, and
- allows improving MA-CBS algorithm through better use of the heuristic.

Consequently, we introduce a new algorithm, Merge-and-Restart Conflict-Based Search (MR-CBS), which differs in the way meta-agents are handled during the search. We empirically compare the new algorithm on different problem domains to illustrate a steady increase of performance.

## Background and Related Work

The pseudocode for MA-CBS is shown in Algorithm 1. Like CBS, MA-CBS maintains a list of nodes, sorted by the increasing sum of costs of individual solutions (SIC). At every step of the main loop (lines 3–15) a node with the lowest SIC is removed from the node list (line 6). If the solutions in the node do not have any conflicts, this set of solutions is returned as the solution for the problem (line 15). In case of conflicts there are two possibilities. MA-CBS either adds, just like CBS, two nodes to the node list. The nodes are created according to a single conflict between a pair of agents. Each of the nodes has the solution for one of the agents updated to avoid the conflict with the other agent (lines 12–14). Otherwise, MA-CBS merges the two agents into a combined agent and adds a single node to the node list with the combined agent instead of the pair of agents (lines 9–10). The decision whether to split or to merge is based on parameter  $B$ : the agents are merged if the number of encountered conflicts between the agents since the beginning of the search is at least  $B$ .

---

### Algorithm 1 MA-CBS

---

```

1: procedure MA-CBS(Agents, B)
2:   Nodelist  $\leftarrow$  [NODE(Agents)]
3:   loop
4:     if EMPTY?(Nodelist) then return FAILURE
5:     else
6:       Node  $\leftarrow$  POP(Nodelist)
7:       if CONFLICTS?(Node) then
8:         if MERGE?(Node, B) then
9:           Node'  $\leftarrow$  MERGE(Node)
10:          INSERT(Node', Nodelist)
11:        else
12:          Node', Node''  $\leftarrow$  SPLIT(Node)
13:          INSERT(Node', Nodelist)
14:          INSERT(Node'', Nodelist)
15:        else return SOLUTIONS(Node)

```

---

Both CBS and MA-CBS solve MAPF optimally, however sub-optimal variants of CBS were also introduced (Barré et al. 2014). On the other hand, different algorithms for solving MAPF optimally are also pursued. Some of the other algorithms bear similarities to MA-CBS, such as Independence Detection (ID) (Standley 2010), which for every pair of conflicting agents tries to find an alternative solution for each

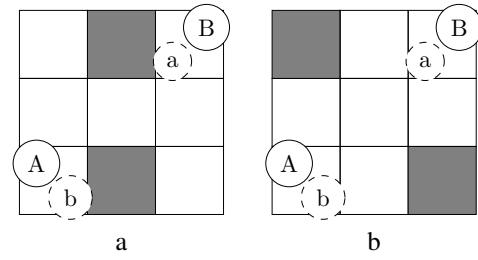


Figure 1: Scenes with 2 agents.

agent avoiding the conflicts, and if failed merges the conflicting agents into a combined agent. A suboptimal variant of ID offers a trade-off between running time and solution quality (Standley and Korf 2011). Other algorithms, such as A\*+OD (Standley 2010), EPEA\* (Felner et al. 2012), or ICTS (Sharon et al. 2013) can be used for lower-level search in MA-CBS.

## Justification of Fixed Threshold

The authors of MA-CBS summarized the results of their empirical evaluation of the algorithm with an evidence that:

- The best value of threshold  $B$  decreases with hardness of the problem instances.
- The advantage of MA-CBS is more prominent on harder instances.

Such behavior is characteristic for *online competitive algorithms* (Manasse, McGeoch, and Sleator 1988), and in particular reminds of the *ski rental problem*. In the ski rental problem a tourist at a ski resort may either pay a fixed rent for each day of ski rental, or to buy the ski, obviously at a higher price. The famous result for this problem is that the tourist should rent the ski for  $\frac{\text{ski price}}{\text{daily rent}} - 1$  days, and to buy the ski on the next day if he/she is still at the resort.

Consequently, we conjectured that the fixed threshold in MA-CBS plays a role similar to the threshold in the online algorithm for the ski rental problem. Both theoretical analysis and empirical evaluation confirmed this conjecture.

## Model Problem: 2 agents

Consider the MAPF problem for 2 agents as the simplest non-trivial case. If MA-CBS is used, and the number of conflicts reaches  $B$ , some number of merges between 1 and the number of nodes currently in the node list solves the problem instance. If the time to find a solution for the combined agent does not become much shorter when constraints are added, it may be better to just remove all constraints and compute the solution for the combined agent *once*, rather than multiple times for each node in the node list. We shall call a version of MA-CBS that restarts the search upon a merge MR-CBS. A comparative evaluation of MA-CBS and MR-CBS is provided in Table 1. The problem instance is shown in Figure 1.a. The number of merges performed by MA-CBS is, for all but extreme, (1 and 8) values of  $B$  is greater than 1 (the number of restarts in MR-CBS), and the number

of single-agent nodes expanded by MA-CBS is greater than by MR-CBS.<sup>2</sup>

| B  | merges | nodes  |        |
|----|--------|--------|--------|
|    |        | MA-CBS | MR-CBS |
| 1  | 1      | 66     | 66     |
| 2  | 2      | 136    | 80     |
| 3  | 3      | 207    | 95     |
| 4  | 4      | 278    | 110    |
| 5  | 3      | 238    | 126    |
| 6  | 2      | 198    | 142    |
| 7  | 1      | 158    | 158    |
| 8+ | 0      | 118    | 118    |

Table 1: MR-CBS vs MA-CBS for scene 1.a.

The intuition behind MR-CBS is formalized by the following two lemmas about competitiveness of both MR-CBS and MA-CBS for 2 agents:

**Lemma 1.** *Let us denote by  $T_2$  the time to find the shortest path for the combined agent, and by  $T_{1,1}$  the time to find the shortest paths for both agents independently, ignoring conflicts between the agents. Under the assumptions that*

- a)  $T_{1,1}$  and  $T_2$  are constant for a given problem instance at any point of the algorithm,
- b)  $T_2 \geq T_{1,1}$ , and
- c) the ratio  $\frac{T_2}{T_{1,1}}$  is known in advance,

MR-CBS is  $2 - \frac{1}{B}$ -competitive, and the competitive ratio is achieved for  $B = \lfloor \frac{T_2}{T_{1,1}} \rfloor$ .

*Proof.* Since merging two agents solves a 2-agent problem at the cost  $T_2$ , and splitting on a conflict may or may not solve the problem at the cost  $T_{1,1}$ , this problem is equivalent to the ski rental or two caches and one block snoopy caching problem (Karlin et al. 1988).  $\square$

**Lemma 2.** *Under the assumptions of Lemma 1 MA-CBS is  $1 + B - \frac{1}{B}$ -competitive, and the competitive ratio is achieved for  $B = \lfloor \frac{T_2}{T_{1,1}} \rfloor$ .*

*Proof.* After  $k$  splits there are  $k + 1$  nodes in the node list (Algorithm 1) for any  $k \geq 0$ . Hence, MA-CBS performs at most  $B - 1$  splits and then at most  $B$  merges, and the worst-case time is  $T_{MA-CBS} = BT_2 + (B - 1)T_{1,1}$ . Just like in the proof for the ski rental problem, the competitive ratio is

$$\min_B \min \left( \frac{T_{MA-CBS}}{T_2}, \frac{T_{MA-CBS}}{BT_{1,1}} \right) = 1 + B - \frac{1}{B} \quad (1)$$

for  $B = \lfloor \frac{T_2}{T_{1,1}} \rfloor$ .  $\square$

<sup>2</sup>Let us note that the number of expanded single-agent nodes is, along with the search time, an adequate measure of the performance of CBS, MA-CBS, and variants. Evaluation of the distance heuristic for a single agent can be memoized, and the total heuristic evaluation time is thus negligible compared to the time spent expanding single-agent nodes and generating children satisfying the constraints.

According to the assumptions of Lemma 1,  $\frac{T_2}{T_{1,1}}$  is at least 1, hence MR-CBS is competitive with a lower ratio (that is, in the worst case finds a solution in a shorter time) than MA-CBS.

The worst-case approach is apparently a reasonable option for designing an algorithm for 2-agent MAPF. Both problem instances in Figure 1 have agents at the same locations, as well as the same number of passable cells, and the same position of the bottleneck. Nonetheless, the cost of an optimal solution for the instance in Figure 1.a is 11, and CBS has to resolve 7 conflicts, but for the instance in Figure 1.b the cost is 9, and only 1 conflict has to be resolved before a solution is found. MR-CBS is more efficient for 1.a but not for 1.b, where CBS is faster.

## MR-CBS for Any Number of Agents

MR-CBS can be extended to an arbitrary number of agents. The pseudocode of MR-CBS is shown in Algorithm 2. MR-CBS differs from MA-CBS (Algorithm 1) in lines 8–10. Firstly, MERGE/R creates a node with unconstrained solutions for individual agents. Secondly, the node list is re-initialized to contain just the new node (line 10). Effectively, the search is restarted with the two agents replaced by a combined agent.

---

### Algorithm 2 MR-CBS

---

```

1: procedure MR-CBS(Agents,B)
2:   Nodelist  $\leftarrow$  [NODE(Agents)]
3:   loop
4:     if EMPTY?(Nodelist) then return FAILURE
5:     else
6:       Node  $\leftarrow$  POP(Nodelist)
7:       if CONFLICTS?(Node) then
8:         if RESTART?(Node, B) then
9:           Node'  $\leftarrow$  MERGE/R(Node)
10:          Nodelist  $\leftarrow$  [Node']
11:        else
12:          Node', Node''  $\leftarrow$  SPLIT(Node)
13:          INSERT(Node', Nodelist)
14:          INSERT(Node'', Nodelist)
15:        else return SOLUTIONS(Node)

```

---

The decision whether to merge two agents and restart the search is again based on a fixed threshold. Given the suggested interpretation for  $B$  as an estimate of  $\frac{T_2}{T_{1,1}}$ , merging *combined*, instead of single, agents into a larger yet agent should require a threshold that depends on the sizes of the agents to be merged. This was confirmed by preliminary experiments on partial sliding tile puzzle (see below), which showed that using *the same*  $B$  for merging both single and combined agents, as in the original version of MA-CBS (Sharon et al. 2012b), *slows down* the search compared to merging just single agents. Indeed, the number of children grows exponentially with the number of single agents in a combined agent, and thus the search time grows at least exponentially with the size of the combined agent, demanding a higher  $B$ . In the experiments, we limited the maximum

size of a combined agent to 2, that is, only single agents would be merged, efficiently setting  $B = \infty$  for producing combined agents consisting of more than 2 single agents. A more advanced implementation would be based on different values of  $B$  for different sizes of agents to be merged.

### Exploring MR-CBS with Partial Sliding Tile Puzzle

We used the *partial* sliding tile puzzle, in which only some of the tiles are present on the  $4 \times 4$  board, for the exploration. We needed to determine the number of tiles for which problem instances which are solved faster by either CBS and MA-CBS(1) or MR-CBS(1) occur sufficiently often. We randomly generated a set of 100 random scenes for every number of agents from 2 to 9, spreading the agents in such a way that conflicts are likely, and found that most instances form with less than 7 agents are best solved by CBS without merging, and most problems with 9 agents are best solved by merging agents on the first conflict. Consequently, we chosen a set of 8-agent problem instances.

| B   | MR-CBS         |                   | MA-CBS    |             |
|-----|----------------|-------------------|-----------|-------------|
|     | time, sec      | expanded          | time, sec | expanded    |
| 1   | 7,997.4        | 35,214,246        | 12,077.5  | 42,593,750  |
| 4   | 9,170.4        | 41,795,136        | 32,443.3  | 136,769,252 |
| 19  | 7,011.4        | 30,990,369        | 43,633.9  | 170,588,470 |
| 63  | 6,249.0        | 25,826,426        | 46,647.9  | 160,939,149 |
| 94  | <b>5,480.9</b> | <b>21,999,664</b> | 48,409.4  | 191,118,297 |
| 211 | 6,940.8        | 28,448,940        | 48,993.2  | 199,920,613 |
| 317 | 7,047.4        | 28,700,663        | 40,329.0  | 162,546,522 |

Table 2: MR-CBS vs. MA-CBS on  $4 \times 4$  tile puzzle, 8 agents.

The relative performance of MR-CBS and MA-CBS is consistent with the results for 2 agents. Table 2 shows the running time, and the number of expanded nodes for MR-CBS and MA-CBS. The advantage of MR-CBS over MA-CBS is more prominent for higher values of  $B$ , where MR-CBS exhibits much lower running times and numbers of expanded nodes. Only by  $B = 317$  the search time and the number of expanded nodes begin to decrease; this, like in the case of 2 agents, can be explained by the decrease in the number of search branches reaching this number of conflicts.

### Experiments on Benchmark Maps

Following the empirical evaluation in (Sharon et al. 2012b), we used the same three maps from the game *Dragon Age: Origins* (Sturtevant 2012). As with the puzzle, 100 random instances were generated, and the reported numbers are the totals over the 100 instances. Table 3 shows the results of CBS, MA-CBS, MR-CBS on 16-agent scenes for a range of values of  $B$ .

Again, MR-CBS shows the best performance (bold in the table). The advantage of MR-CBS over MA-CBS depends on the map. **den520d** consists mostly of open spaces, and MR-CBS is **5 times** faster than MA-CBS for the tested values of  $B$ . **ost003d** is a combination of open spaces and bottlenecks; MR-CBS (for  $B = 16$ ) is 10% faster than MA-




|             |  <b>den520d</b> |  <b>ost003d</b> |  <b>brc202d</b> |
|-------------|--|--|--|
| CBS         | 68,321   | 9,800  | 174,727  |
| MR-CBS(1)   | 3,364  | 7,429  | 97,442   |
| MA-CBS(1)   | 2,935  | 7,351  | 98,455   |
| MR-CBS(16)  | <b>707</b>   | <b>6,837</b>   | <b>66,286</b>  |
| MA-CBS(16)  | 3,531  | 26,561   | 75,212   |
| MR-CBS(64)  | 804  | 8,373  | 72,417   |
| MA-CBS(64)  | 6905   | 30,582   | 78,430   |
| MR-CBS(256) | 1,454  | 10,296   | 92,086   |
| MA-CBS(256) | 18,704   | 17,844   | 99,987   |

Table 3: *Dragon Age: Origins* scenes with 16 agents, search times (sec) for CBS, MR-CBS, MA-CBS.

CBS (for  $B = 1$ ) for the tested values of  $B$ , but for intermediate values of  $B$ , 16 and 64, MR-CBS is **4 times** faster than MA-CBS: hence the best value that would be estimated from test runs of the low-level search on single and combined agents might give a 4-fold increase in performance. **brc202d** mostly consists of narrow paths resulting in many bottlenecks. MR-CBS is  $\approx 12\%$  faster than MA-CBS for the tested values of  $B$ . For all maps and for all values of  $B$ , MR-CBS is faster than MA-CBS. Moreover, the search time of MA-CBS for intermediate values of  $B$  is often longer than for extreme (either low or high) values, an evidence which further supports the advantage of restarting the search upon a merge. Delayed or Randomized MR-CBS can be used to further improve the performance for the best found value of  $B$ .

### Summary and Future Work

This paper has several contributions:

- We provided a justification for the use of a fixed threshold for decision-making in MA-CBS; the justification was based on the worst-case analysis of a two-agent MAPF problem.
- Using a model problem based on this justification we introduced a new algorithm, MR-CBS, where the search is restarted after a merge. MR-CBS exhibits shorter search times and lower numbers of expanded nodes than MA-CBS on both  $4 \times 4$  partial sliding tile puzzle and computer game scenes.

There is room for further improvement of MR-CBS. Firstly, the decision to merge a pair of agents can be made based on the history of conflict occurrence and resolution through splitting, rather than just the number of conflicts. Secondly, the tie-breaking, such as selection of the conflicting agents to split on or merge, and of the conflict to resolve in case of a split, can be improved using heuristic decision rules. We believe that metareasoning techniques (Russell and Wefald 1991; Russell 2014) can be applied successfully to MAPF domain in general and MA-CBS variants in particular to design the heuristics.

## References

- [Barrer et al. 2014] Barrer, M.; Sharon, G.; Stern, R.; and Felner, A. 2014. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In *Proceedings of the Seventh Annual Symposium on Combinatorial Search, SOCS 2014, Prague, Czech Republic, 15-17 August 2014*.
- [Felner et al. 2012] Felner, A.; Goldenberg, M.; Sharon, G.; Stern, R.; Beja, T.; Sturtevant, N. R.; Schaeffer, J.; and Holte, R. 2012. Partial-expansion a\* with selective node generation. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*.
- [Hernavolygi and Holte 2004] Hernavolygi, I., and Holte, R. C. 2004. Steps towards the automatic creation of search heuristics. Technical Report TR04-02, Computer Science Department, University of Alberta.
- [Karlin et al. 1988] Karlin, A. R.; Manasse, M. S.; Rudolph, L.; and Sleator, D. D. 1988. Competitive snoopy caching. *Algorithmica* 3:77–119.
- [Manasse, McGeoch, and Sleator 1988] Manasse, M. S.; McGeoch, L. A.; and Sleator, D. D. 1988. Competitive algorithms for on-line problems. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA, 322–333*.
- [Prieditis 1993] Prieditis, A. 1993. Machine discovery of effective admissible heuristics. *Machine Learning* 12:117–141.
- [Ratner and Warmuth 1986] Ratner, D., and Warmuth, M. K. 1986. Finding a shortest solution for the  $N \times N$  extension of the 15-puzzle is intractable. In *Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia, PA, August 11-15, 1986. Volume 1: Science.*, 168–172.
- [Russell and Wefald 1991] Russell, S., and Wefald, E. 1991. Principles of metereasoning. *Artificial Intelligence* 49:361–395.
- [Russell 2014] Russell, S. 2014. Rationality and intelligence: A brief update. In *Vincent C. Muller (ed.), Fundamental Issues of Artificial Intelligence (Synthese Library)*. Berlin: Springer. To appear.
- [Sharon et al. 2012a] Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R. 2012a. Conflict-based search for optimal multi-agent path finding. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*.
- [Sharon et al. 2012b] Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R. 2012b. Meta-agent conflict-based search for optimal multi-agent path finding. In *SOCS*.
- [Sharon et al. 2013] Sharon, G.; Stern, R.; Goldenberg, M.; and Felner, A. 2013. The increasing cost tree search for optimal multi-agent pathfinding. *Artif. Intell.* 195:470–495.
- [Silver 2005] Silver, D. 2005. Cooperative pathfinding. In *AIIDE*, 117–122.
- [Standley and Korf 2011] Standley, T. S., and Korf, R. E. 2011. Complete algorithms for cooperative pathfinding problems. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, 668–673.
- [Standley 2010] Standley, T. S. 2010. Finding optimal solutions to cooperative pathfinding problems. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*.
- [Sturtevant 2012] Sturtevant, N. R. 2012. Benchmarks for grid-based pathfinding. *IEEE Trans. Comput. Intellig. and AI in Games* 4(2):144–148.
- [Wang, Botea, and Kilby 2011] Wang, K.-H. C.; Botea, A.; and Kilby, P. 2011. On improving the quality of solutions in large-scale cooperative multi-agent pathfinding. In *SOCS*.