Effectively Incorporating Weighted Cost-to-go Heuristic in Suboptimal CBS

Rishi Veerapaneni, Tushar Kusnur, Maxim Likhachev

Robotics Institute, Carnegie Mellon University
{rveerapa, tkusnur, mlikhach}@andrew.cmu.edu

Abstract

Conflict-Based Search (CBS) is a popular multi-agent path finding (MAPF) solver that employs a low-level single agent planner and a high-level constraint tree to resolve conflicts. The vast majority of modern MAPF solvers focus on improving CBS by reducing the size of this tree through various strategies with few methods modifying the low level planner. All low level planners in existing CBS methods use an unweighted cost-to-go heuristic, with suboptimal CBS methods also using a conflict heuristic to help the high level search. Contrary to prevailing beliefs, we show that the cost-to-go heuristic can be used significantly more effectively by weighting it in a specific manner alongside the conflict heuristic. We introduce two variants of doing so and demonstrate that this change can lead to 2-100x speedups in certain scenarios. Additionally, to the best of our knowledge, we show the first theoretical relation of prioritized planning and bounded suboptimal CBS and demonstrate that our methods are their natural generalization.

1 Introduction

Multi-Agent Path Finding (MAPF) is the problem of computing collision-free paths for a team of agents in a known environment while minimizing a measure of their travel times. This is required for several real-world tasks such as the smooth operation of automated warehouses (Li et al. 2020b), robot soccer (Biswas et al. 2014), collaborative manufacturing (Sun and Mills 2002), coverage (Kusnur et al. 2021), and others. MAPF is a challenging problem and is shown to be NP-complete (Ratner and Warmuth 1986).

Prioritized Planning (PP) (Erdmann and Lozano-Perez 1987) is a fast multi-agent planning approach that sequentially plans agents avoiding earlier agents with better "priority", and has been applied to several domains (Wu, Bhattacharya, and Prorok 2020; Čáp et al. 2015; Velagapudi, Sycara, and Scerri 2010). However PP provides no guarantees on completeness or bounded suboptimality.

Conflict-Based Search (CBS) is a popular complete and optimal MAPF solver that employs a low-level single agent planner and a high-level constraint tree (CT) to resolve conflicts. Several methods speed up CBS by reducing the CT size by explicitly pruning branches, selectively expanding branches, adding sets of constraints, detecting symmetries, and improving high-level heuristics (Boyarski et al. 2015, 2021; Li et al. 2019, 2020a, 2021).

Enhanced CBS (ECBS) (Barer et al. 2014) introduced the first bounded-suboptimal version of CBS, utilizing a focal search on the high level as well as another focal search planner on the low level that minimizes path conflicts with other agents and therefore decreases the CT size. ECBS specifically mentions how modifying the low level planner to use a weighted cost-to-go heuristic returns paths with many conflicts, leading to a larger CT tree and proved "ineffective in [their] experiments" as direct motivation for reducing the path conflits instead. Explicit Estimation CBS (EECBS) (Li, Ruml, and Koenig 2021) replaces ECBS's high level focal search with Explicit Estimation Search (Thayer and Ruml 2011) but keeps the same low level focal search. Continuous-time CBS (CCBS) (Andreychuk et al. 2019) incorporates Safe Interval Path Planning (SIPP) (Phillips and Likhachev 2011) to speed up the low level search by reasoning about waits but also does not employ a weighted heuristic. To the authors' best knowledge, no prior work has effectively used a weighted cost-to-go heuristic in any manner in the CBS framework, with the prevailing norm that doing so would lead to more conflicts, reduce performance, or remove bounded sub-optimality.

Our key insight is that we can use the conflict heuristic *along with a weighted cost-to-go heuristic*. To the authors' knowledge, we introduce the first bounded sub-optimal CBS methods that incorporates a weighted cost-to-go heuristic with the conflict heuristic within CBS's single agent planner. Our contributions are

1. Incorporating the weighted cost-to-go heuristic in the open queue, and studying how the path lower bounds interact with certain CBS improvements.
2. Combining the weighted cost-to-go heuristic with a weighted conflict heuristic in the focal queue, and discovering an important relationship between the two weights.
3. Reducing PP to one particular step of sub-optimal CBS and showing that our weighted variant is the natural generalization. We recommend practitioners using PP switch to our methods as they can get the same initial prioritized planning behaviour while still being complete.

2 Incorporating Weighted Cost-to-go Heuristic

CBS utilizes an optimal space-time A* low level planner with a precomputed cost-to-go heuristic that measures the optimal distance to goal ignoring conflicts. Bounded sub-optimal CBS methods (e.g. ECBS, EECBS) modify the single agent planner to a focal search that computes w_{so} sub-optimal path that minimizes the number of conflicts with other agents (which reduces future constraints in the CT). The low level planner must also return a lower bound on the optimal solution cost which is required for certain CBS improvements, specifically prioritized conflicts and symmetry reasoning, see Li, Ruml, and Koenig (2021) for full justification. The low level focal search has two queues; OPEN which searches over optimal paths (paths sorted by cost) and maintains an optimality bound, and FOCAL which prioritizes w_{so} sub-optimal paths with fewer conflicts (paths sorted by conflicts). We specifically discuss our method in relation to EECBS as it was shown to outperform ECBS and other MAPF planners, but our method is directly usable in ECBS and any other bounded sub-optimal CBS planner using a low level focal planner (see Table 4).

Our main idea is to incorporate a weighted cost-to-go heuristic in the single agent planner in two ways: one in OPEN independent of the conflict heuristic and the other in FOCAL along with the conflict heuristic. Algorithm 1 showcases EECBS's general low level search pseudocode, with functions W-EECBS changes highlighted in blue. Ties in FOCAL are broken by f_{open}. The user's sub-optimality hyper-parameter w_{so} is assumed to be fixed and outside our optimization.

Algorithm 1: Suboptimal CBS low level focal search planner

Input: n_{start}, atGoal(), Paths P_I of other agents
Output: Lower bound LB on optimal path cost, Path from n_{start} with sub-optimality $\leq w_{so}$ (i.e. cost $\leq w_{so} * LB$)

SetW_f()
OPEN = FOCAL = $\{n_{start}\}$, $LB = F_{best} = 0$
while FOCAL $\neq \emptyset$ **do**
  $n \leftarrow$ FOCAL.pop()
  OPEN.remove(n)
  $LB \leftarrow \max(LB,$ UpdateLowerBound()$)$
  if atGoal(n) **then**
    return LB, Solution backtracking from n
  for $n' \in$ succ(n) **do**
    $g \leftarrow n.g + cost(n, n')$
    $h \leftarrow$ getCostToGoHeuristic(n')
    $n'.F_{open} \leftarrow f_{open}(g, h)$
    OPEN.insert(n')
    $c \leftarrow$ getNumConflictsFromPaths(n', P_I)
    $n'.F_{focal} \leftarrow f_{focal}(g, h, c)$
  ▷ Update FOCAL
  $F_{best} \leftarrow min_{k \in OPEN}\ k.F_{open}$
  for all $n' \in OPEN, \notin FOCAL$ **do**
    if $n'.F_{open} \leq w_f * F_{best}$ **then**
      FOCAL.insert(n')
return NaN, No solution

procedure $f_{open}(g, h)$:
  return $g + h$
procedure $f_{focal}(g, h, c)$:
  return c
procedure SETW_f():
  $w_f \leftarrow w_{so}$
procedure UPDATELOWERBOUND():
  return F_{best}

Weighted Open Variant (WO-EECBS)

OPEN's priority function is weighted by w_h, while FOCAL remains unchanged, prioritized by the number of conflicts. To maintain our overall suboptimality bound, the focal bound w_f is scaled to w_{so}/w_h which constrains $w_h \in [1, w_{so}]$ as we need $w_f \geq 1$. Since the f-values in OPEN are now weighted by w_h, we obtain a lower bound on the optimal path cost by scaling the minimum f-value in OPEN, F_{best}, to F_{best}/w_h. Note that $w_h = 1$ trivially results in regular EECBS.

One side effect of this method is that this naively computed lower bound is usually substantially lower than the optimal path cost even though the path may not have been very sub-optimal. Several papers have discussed this pessimistic lower bound in weighted A* single agent search (Thayer and Ruml; Holte et al. 2019). This pessimistic lower bound should then theoretically reduce the amount of prioritized conflicts (PC) and symmetry reasoning (SR) applied. We therefore *a posteriori* compute a better lower bound using Holte et al. (2019) and test if this increases the usage of PC and SR, and boosts performance.

Weighted Focal Variant (WF-EECBS)

We keep OPEN unweighted and instead incorporate the weighted heuristic in FOCAL along with the inadmissible conflict heuristic. This requires us to balance the importance of these competing heuristics via FOCAL's priority function $g + w_h * h + w_c * c$ with $w_h \geq 1, w_c \geq 0$. Changing the magnitude of w_h, w_c changes the relative importance of finding a solution fast (higher w_h) vs avoiding conflicts (higher w_c). Note that $w_h = 1$ and $w_c \rightarrow \infty$ results in regular EECBS (preferring paths with lowest conflicts). Due to the use of FOCAL, w_h can be arbitrarily large and is not bounded by w_{so}. In our experiments we see that WF-EECBS outperforms WO-EECBS and EECBS, therefore Weighted EECBS (W-EECBS) refers to this weighted focal version.

Lemma 1. *WO-EECBS and WF-EECBS are both w_{so} sub-optimal.*

Proof. EECBS's overall optimality is split between the high-level CT sub-optimal search and the low-level sub-optimal search. Since the high-level search is unchanged and identical to EECBS, we just need to prove that WO-EECBS and WF-EECBS have the same low-level sub-optimality w_{so} as EECBS.

In WO-EECBS: FOCAL returns a node at most w_f sub-optimal compared to OPEN which is weighted by w_h. Our

Algorithm 2: Weighted Open modifications

Parameters: Heuristic weight w_h

procedure $f_{open}(g, h)$:
  return $g + w_h * h$
procedure $\text{SET}W_f()$:
  $w_f \leftarrow w_{so}/w_h$
procedure GETLOWERBOUND(): ▷ Naive
  return F_{best}/w_h
procedure GETLOWERBOUND(): ▷ Improved
  $g_{min} \leftarrow min_{n \in OPEN}\ n.g$
  return $(F_{best} + (w_h - 1) * g_{min})/w_h$

Algorithm 3: Weighted Focal modifications

Parameters: Heuristic weight w_h, Conflict weight w_c

procedure $f_{focal}(g, h, c)$:
  return $g + w_h * h + w_c * c$

overall optimality is then $w_f * w_h = w_{so}/w_h * w_h = w_{so}$.

In WF-EECBS: FOCAL's sub-optimality is fixed regardless of f_{focal}, and OPEN is optimal, so our overall optimality is trivially $w_f = w_{so}$. □

Relating CBS, Prioritized Planning, and W-EECBS

CBS-based algorithms and PP are usually treated as distinct categories of MAPF search based methods. Ma et al. (2019) introduces priorities in CBS as a distinction to regular CBS and Li et al. (2022) employs a modified PP planner that return paths with least conflicts, but neither attempt to relate PP and CBS.

Here we prove that PP is actually equivalent to the first step of generating the initial agent paths in the root CT node in EECBS (and other bounded sub-optimal CBS planners like ECBS) with an infinite sub-optimality. With $w_{so} = \infty$ in EECBS, all states in OPEN in the single agent planner are inserted into FOCAL, and therefore expansions are sorted first by their number of conflicts, and then the path f-value. In the root CT node, agents will try to avoid all previous agents and search over all conflict=0 paths, then conflict=1 after exhausting all conflict=0 paths, then conflict=2, etc. This first step is identical to PP; EECBS with $w_{so} = \infty$ differs only in its ability to continue planning over conflicts while PP fails in that scenario. To the authors' knowledge, this is the first time there has been an explicit relation between sub-optimal CBS and PP. WO-EECBS and WF-EECBS are the two generalized methods combining the weighted low-level planner commonly used in PP with EECBS's conflict resolution mechanism.

3 Experimental results

We test our methods with different numbers of agents, in increments of 50, on 8 diverse maps (titled in each plot) from Stern et al. (2019) and report the mean values across 5 seeds. Table 1 shows the diversity of the 8 maps; each plot contains the 8 maps in the same order sorted by decreasing number of free states.

Map name	Max # agents	Raw Area	# free states
Paris_1_256	1000	256x256	47240
den520d	1000	256x257	28178
ht_chantry	1000	162x141	7461
den312d	1000	65x81	2445
empty-48-48	1000	48x48	2304
empty-32-32	500	32x32	1024
random-32-32-10	450	32x32	922
random-32-32-20	400	32x32	819

Table 1: **Map statistics —** We show the maximum number of agents, height by width raw area, and number of free states on each of the eight maps that we use to evaluate our methods. Figures are sorted in this same order horizontally (top left subplot will be the largest map, bottom right will be the smallest), to showcase the relationship of performance with map size.

We use $w_{so} = 2$ and a timeout of 300 seconds in all our experiments unless otherwise specified. In all figures, if a method failed (timed out on all 5 seeds) on a particle number of agents on a map, we do not report larger number of agents, see Appendix Section B for full justification. The speed up $S_{method} = T_{baseline}/T_{method}$ is reported to normalize differences in hardware, where the baseline is EECBS. In all tables, speeds up are computed only on instances where the baseline did not timeout.

We provide a short summary of each figure:

Figure 1: Showcases how improving the lower bound increases the utilization of cardinal conflicts and symmetry reasoning in WO-EECBS.

Figure 2: Demonstrates WO-EECBS's varying speed ups across several maps, as well as shows the surprising negative effect of improving the lower bound on overall performance.

Figure 3: Reveals that the ratio between the conflict and cost-to-go weight parameters dictates performance in WF-EECBS.

Figure 4: Highlights WF-EECBS's large performance gains on large maps, and how weighting the cost-to-go heuristic generally helps obtains larger speed ups in larger maps and low-medium conflict regimes (which matches expectations as in high conflict regimes the cost-to-go heuristic will be less informative).

Figure 5: Validates how WO-EECBS with a very large sub-optimality is equivalent to prioritized planning in the root node except with EECBS's conflict resolution mechanism.

Figure 6: Exhibits the increased success rate of using W-EECBS with a very large sub-optimality over prioritized planning, due to W-EECBS ability to resolve conflicts in high conflict regimes.

Weighted Open

Overall, performance with the weighted anchor variant is very varied based on the map; it provides large speed ups (10+) in 2, medium (1-5) in 3, and hurts (0-1) in 3. Figure 1 show that improving the lower bound on the usage of CBS improvements does lead to higher utilization. Contrary to our intuition, Table 2 and Figure 2 reveal this results in worse performance even though this computation has negli-

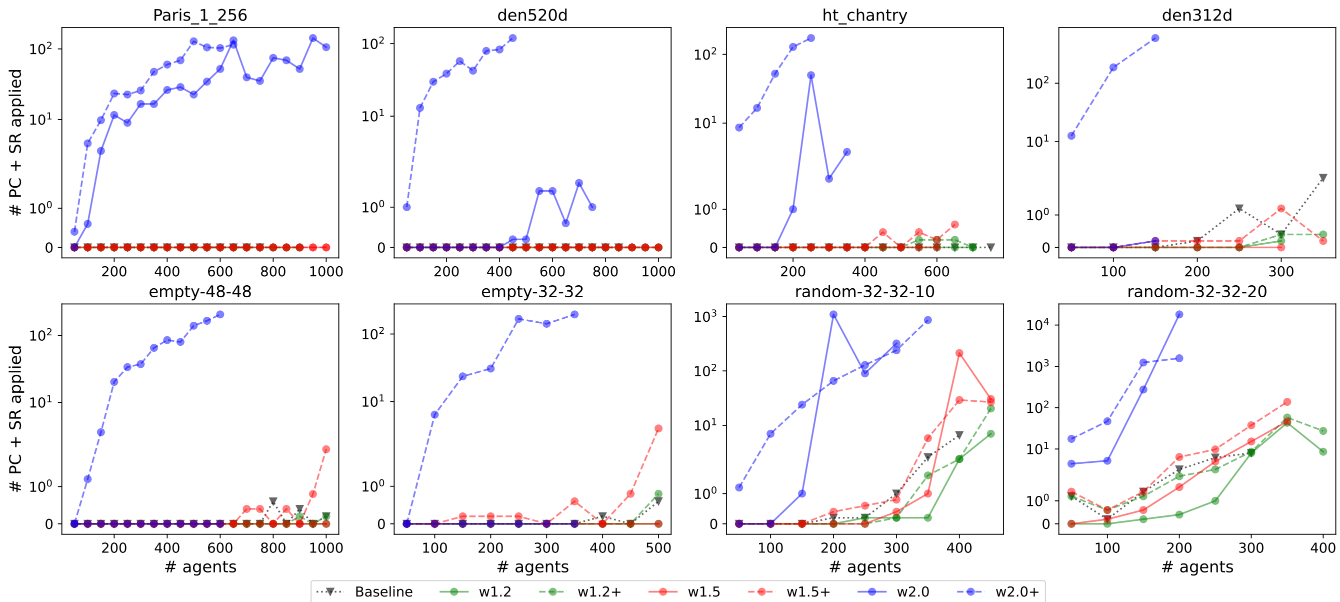

Figure 1: **Viewing the effect of improving the lowerbound on the WO-EECBS —** The "+" label denotes using an improved lowerbound; improving the lowerbound leads to a significant higher usage of CBS improvements with the y-axis denoting the average number (across 5 seeds) of cardinal conflicts and symmetry reasoning applied for each problem instance. Without the improved lower bound, WO-EECBS is usually unable to use these CBS improvements. Methods terminate on a map once they fail all 5 seeds on a certain number of agents or they reach the maximum number of agents in a scene, fractional values are due to the averaging over 5 seeds.

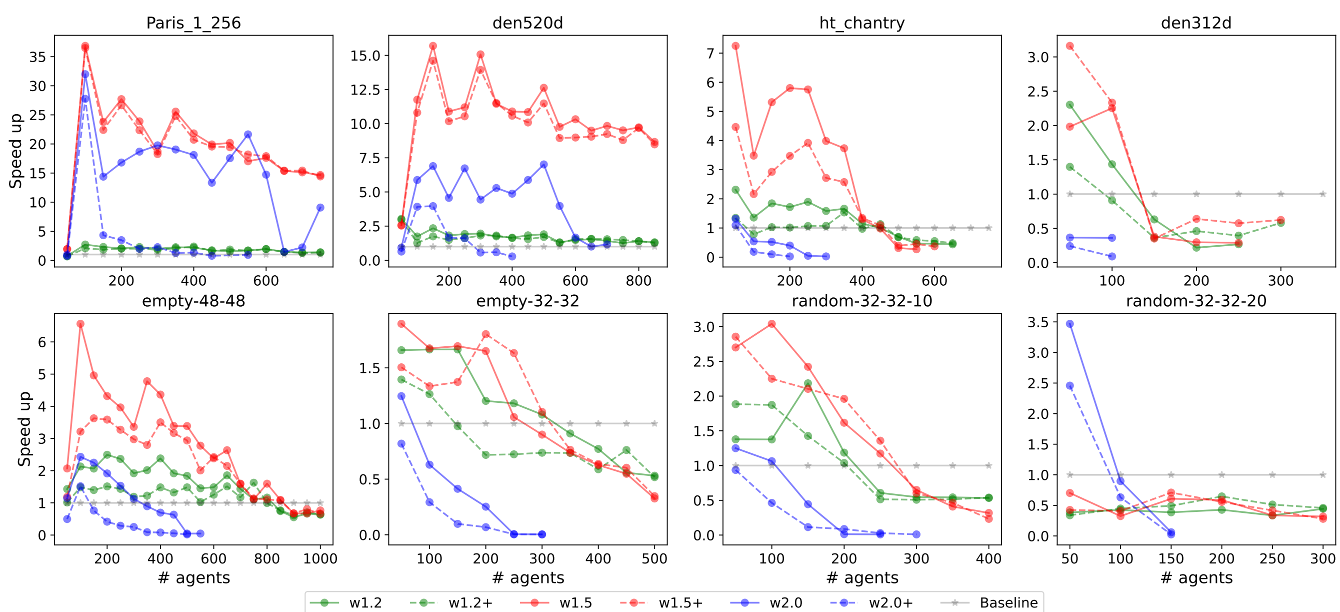

Figure 2: **WO-EECBS full results —** The "+" label denotes using an improved lowerbound. A medium weighted value of w=1.5 performs the best on both maps. However, the maximum speed up peaks at around 35 in the Paris scenario and it struggles in all the harder scenarios (e.g. both random maps, empty-32-32, den312d). Additionally the improved lowerbound actually decreases performance contrary to our intuition.

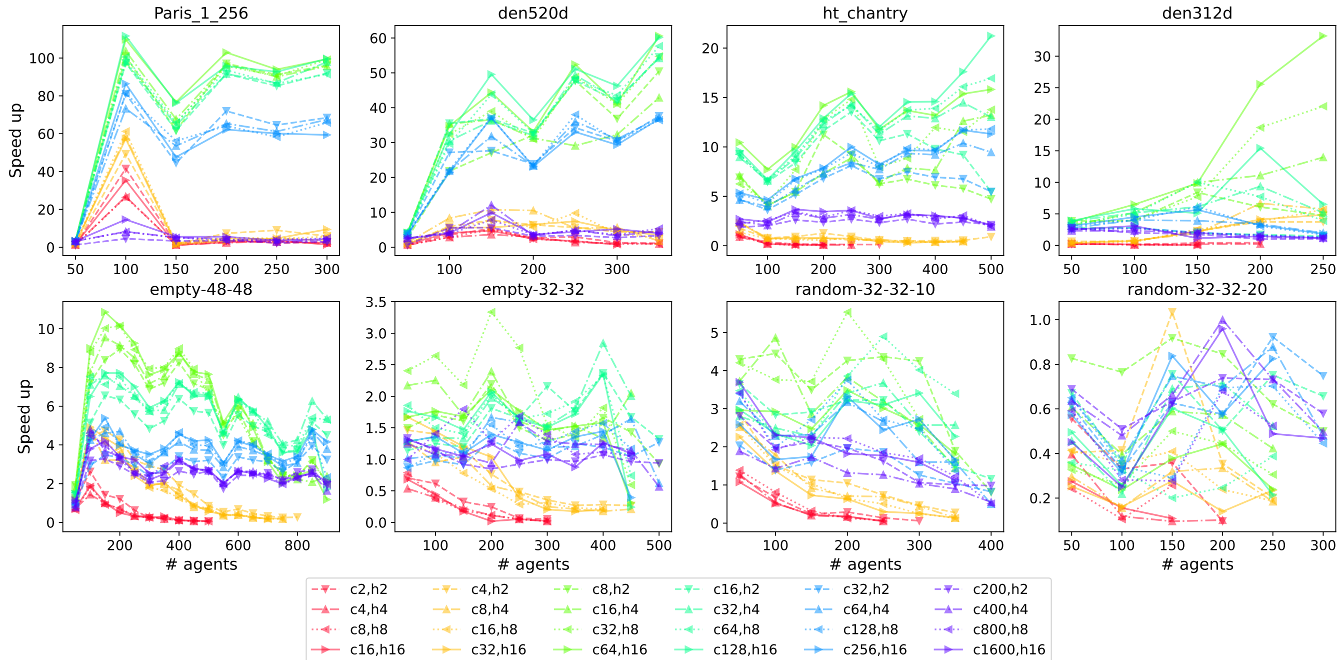

Figure 3: **WF-EECBS Ratio Analysis —** Colors represent different ratios $r = w_c/w_h$ while markers and linestyle denote different w_h values. One would initially think that w_h would cause most of the speed-up while w_c or w_c/w_h plays a secondary effect. The colored bands show the opposite! Performance is tightly linked to r across many w_c and w_h values across all maps. Additionally, values too low $r = 1, 2$ (red, yellow) and too high $r = 16, 100$ (blue, purple) perform worse than $r = 4, 8$ (lime green, turquoise), implying some optimal region of $r \in [2, 16]$. We find in Figure 4 that $r = 5$ works well across most maps.

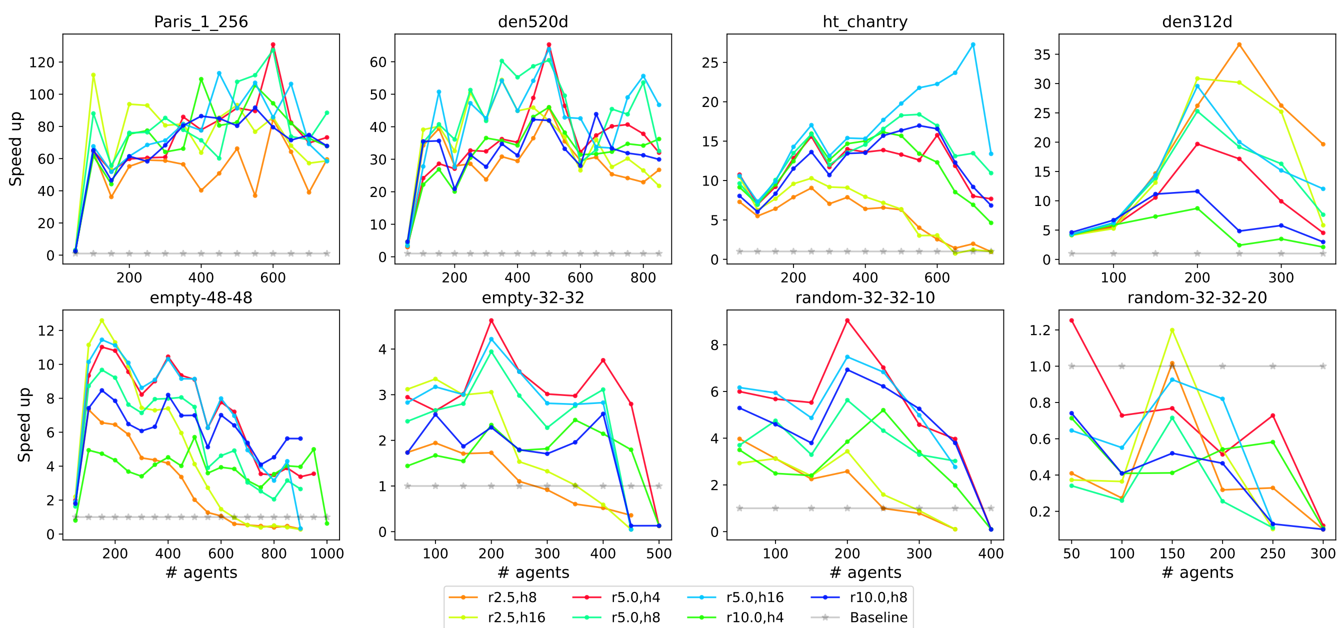

Figure 4: **WF-EECBS Results —** WF-EECBS produces a speed-up factor of 10 or higher on half the maps, and a smaller speed up on three, while performing worse than the baseline on just the random-32-32-20 map. Note the changes in y-axis.

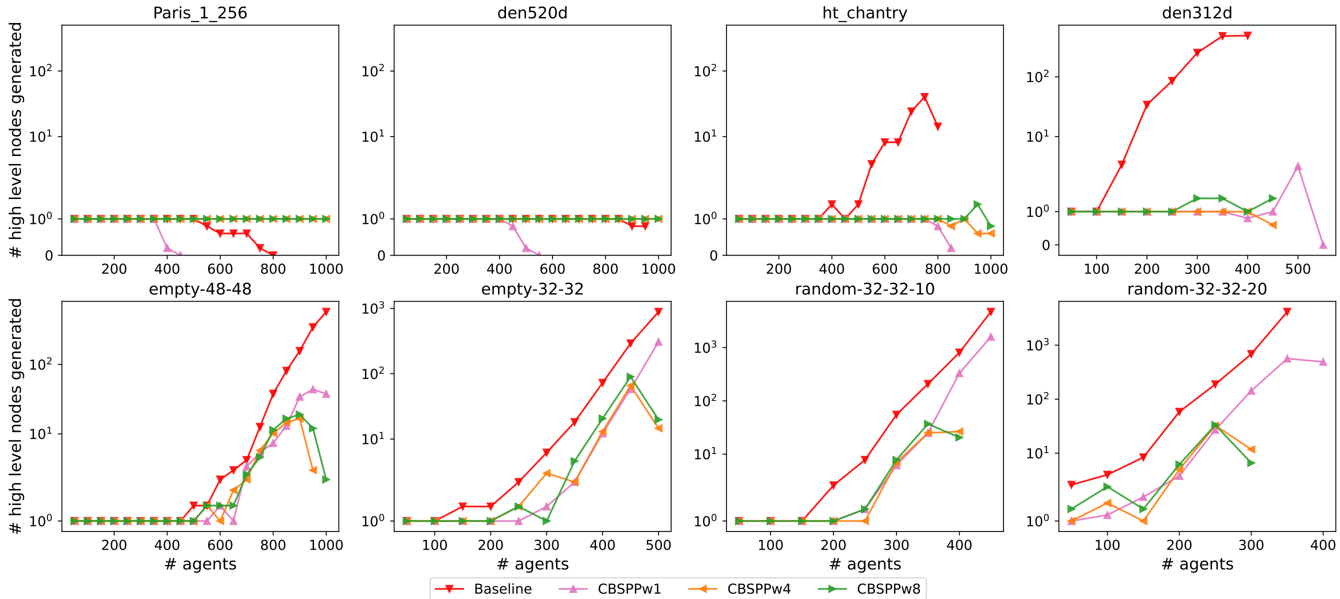

Figure 5: **PP, CBS, W-EECBS equivalence —** We see that running "CBSPP" (WO-EECBS with a very large suboptimality factor simulating $w_{so} \leftarrow \infty, w_h = w$, or equivalently WF-EECBS with $w_{so}, r \leftarrow \infty, w_h = w$) is equivalent to running w weighted prioritized planning with CBS's conflict resolution capability. This is highlighted by the number of high level nodes sticking to one with low numbers of agents (identical to PP) as opposed to the baseline with several high level nodes, and then increasing only after conflicts are forced. Observe how the larger maps (top row) are able to be solved in only one high level node (i.e. no conflict resolution required), but smaller maps require reasoning over conflicts. Note the differing log y-axis values across different graphs. Fractional values due to averaging across 5 seeds.

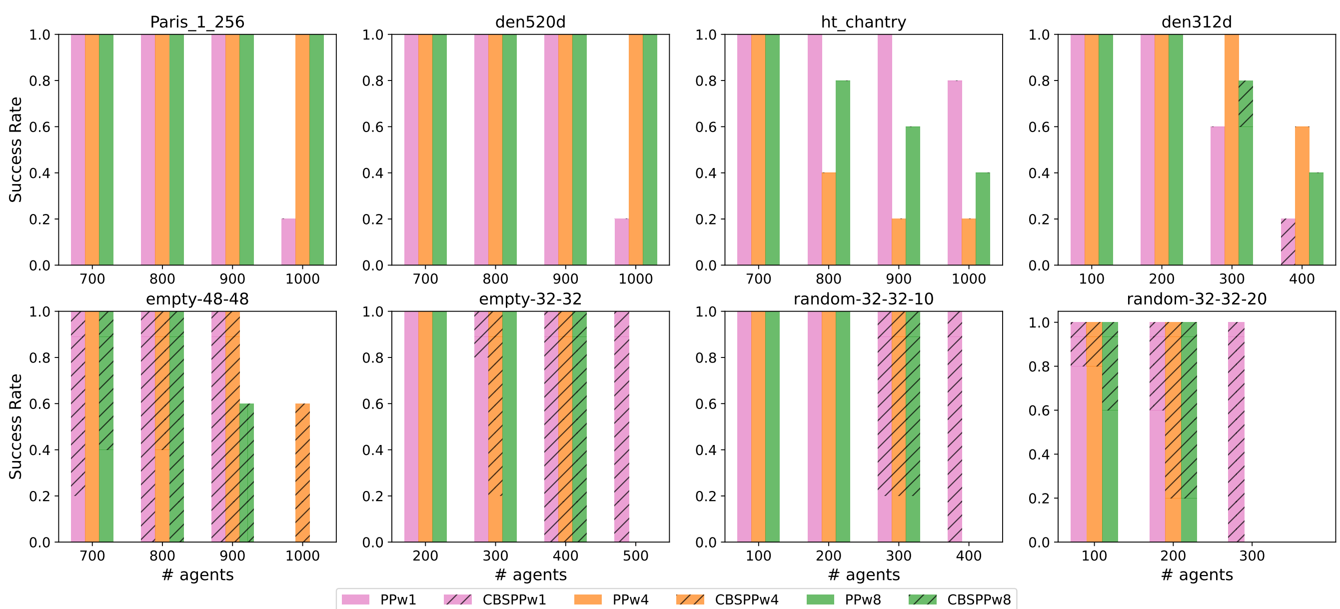

Figure 6: **PP vs CBSPP Success rate —** The increased success rate of the shaded region (CBSPP) over solid (PP) across different weights show the benefit of using EECBS's conflict resolution in high agent regimes or small maps where conflicts become unavoidable. Again observe how the larger maps (top row) are able to be solved with PP (i.e. no conflict resolution required), but smaller maps require reasoning over conflicts.

gible overhead. The respective performance of the weights fits our expectations, larger will help to a certain extent and then hurt due to the interplay of the focal queue. Concretely, WO-EECBS with a "saturated" anchor weight of 2 provides lower speed-ups due to the fact that the focal queue in that instance has $w_f = w_{so}/w_{so} = 1$ and has no flexibility to reduce the number of collisions. The next section demonstrates that this variant is dominated by the weighted focal variant.

Method		Speed up		% instances faster than Baseline	# solved
w_h	LB+	Max	Median		
1	False	1	1	N/A, is baseline	98
1.2	False	3	1.4	69%	102
1.2	True	3	1.2	61%	**103**
1.5	False	**37**	**2.7**	70%	101
1.5	True	36	2.4	**72%**	**103**
2	False	32	0.66	39%	67
2	True	28	0.31	14%	49

Table 2: **WO-EECBS Results —** We report the max and median speed up across 8 maps, as well as the number of instances solved and better than the baseline. We see that $w_h = 1.5$ produces the best speed up and that in general improving the lower bound (LB+ set to True) decreases performance.

Method		Speed up		% faster than Baseline	# solved
r	w_h	Max	Median		
∞	1	1	1	-	98
2.5	8	84	5.5	77%	106
2.5	16	112	6.8	79%	104
5	4	**131**	9.5	**92%**	**113**
5	8	127	9.4	89%	109
5	16	113	**11**	88%	109
10	4	109	5.1	90%	111
10	8	91	7.2	89%	110

Table 3: **WF-EECBS Results —** Comparing against Table 2 we see that WF-EECBS greatly outperforms WO-EECBS and the baseline in the majority of instances. The first row describes the EECBS baseline in WF-EECBS parameters.

Weighted Focal

Table 3 demonstrates that WF-EECBS's speed up is consistently higher than the baseline and WO-EECBS. Overall WF-EECBS helps on 7 out of 8 maps, providing large speed ups (10+) on three and massive speed ups (50-100+) on two. Weighted EECBS (W-EECBS) therefore refers to this weighted focal version.

Figure 3 show a surprising and important relationship between the collision weight w_c and cost-to-go weight w_h in WF-EECBS; the performance is dominated by the ratio $r = w_c/w_h$ rather than the actual w_c or w_h weights, with optimal values $r \in [2, 16]$. The ratio r explicitly dictates the tradeoff between planning longer to avoid a future conflict (collision) or planning shorter and incurring the collision which will need to be resolved by the constraint tree afterwards. Regular EECBS lacks this flexibility and with $r \to \infty$ will prioritize planning longer to avoid conflicts. *To highlight the importance of r, we reparameterize WF-EECBS in respect to r and w_h with $f_{focal}(g, h, c) = g + w_h * (h + r * c)$.* Table 3 shows that increasing w_h with the same r usually but not necessarily increases median speed up.

Figure 4 shows the affect of weighting the cost-go-heuristic w_h based on maps; weighting helps more as the map size increases (starting on the bottom right, see y-axis changes as the maps get bigger). Additionally, for each of the smaller maps (bottom row), relative performance usually decreases as the number of agents increases. Both of these patterns fit our intuition; weighting the cost-to-go heuristic is more useful when paths are longer (larger maps) and less effective when there is more congestion (which would likely cause deviations from the heuristic).

Method		Speedup		% faster than Baseline	# solved vs Baseline
-CBS	w_{so}	Max	Median		
E	1.01	1.7	0.45	23%	30/42
EE	1.01	1.8	0.71	32%	36/46
E	1.1	9.9	2.6	75%	71/76
EE	1.1	10	2.3	73%	68/75
E	1.2	16	3.7	80%	83/77
EE	1.2	22	3.1	80%	79/77
E	1.5	35	3.5	83%	95/73
EE	1.5	47	3.7	78%	94/73
E	2	88	5.5	85%	108/77
EE	2	88	5.5	85%	105/77
E	4	137	7.8	91%	110/66
EE	4	137	8.5	92%	110/67
E	8	164	6.9	91%	112/69
EE	8	164	9.6	91%	111/68

Table 4: **Generalizing weighting FOCAL to different suboptimal CBS methods and suboptimalities. —** We compare the effect of weighting FOCAL on both ECBS and EECBS across different suboptimalities. We use $r = 5, h = 8$ and a timeout of 60 seconds across all experiments, and report statistics as in Table 2. The last column shows the number of instances solved (numerator) vs the baseline (denominator). We see that incorporating the weighted FOCAL hurts at a very low suboptimality $w_{so} = 1.01$, but then produces large benefits for $w_{so} \geq 1.5$. Additionally, we see very similar speedups across different methods at the same suboptimality, demonstrating how our method's benefits are generalizable across different suboptimal CBS methods.

We additionally check how incorporating the weighted heuristic generalizes across different methods and suboptimalities. Table 4 shows the effect of using a weighted focal with $r = 5, h = 8$ across different sub-optimalities on ECBS and EECBS. These hyper-parameters were chosen solely based on Table 3 (on EECBS with $w_{so} = 2$) and were intentionally not optimized for this experiment to see if our weighted heuristic hyper-parameters generalized well. We employ a timeout of 60 seconds and compare against the corresponding unweighted baseline (with the corresponding sub-optimality). We see from both tables that the weighted heuristic hurts at a very low suboptimality $w_{so} = 1.01$ but then steadily results in larger performance boosts as w_{so} increases. In particular, for large suboptimalities starting at $w_{so} = 1.5$, we see that our weighted methods start to solve significantly more instances than the baseline. For $w_{so} \geq 2$, we see that we also get large and consistent speed up benefits ($> 80\%$ faster than baseline, median speed up > 5). For large suboptimalities $w_{so} = 4, 8$, our method is able to solve almost double the number of instances as their unweighted baseline. It is important to observe that our method produces very similar speeds up in both ECBS and EECBS. This highlights how our method identically speeds up the low level planner regardless of the high level search, demonstrating how our technique is readily generalizable to other existing and future sub-optimal CBS methods.

Relating CBS, Prioritized Planning, and W-EECBS

We run WO-EECBS with a very large sub-optimality value ($w_{so} = 10000$) and different anchor weights to see how this mimics running weighted prioritized planning. We denote these as "CBSPP" with their specific weights to emphasize the relation. Figure 5 verifies that the number of generated nodes stays at 1 for low levels of agents until conflicts become unavoidable. Figure 6 demonstrates how CBSPP's ability to replan using CBS's conflict resolution increases success rate compared to prioritized planning. When possible, practitioners using PP should instead use W-EECBS with a large sub-optimality as they get the same prioritized planning behavior in the root node along with the natural robustness and completeness of CBS.

4 Conclusion and Future Work

We see several avenues to directly build upon our work. Our work keeps r and w_h fixed in MAPF instances; adaptively changing r and w_h during a single MAPF search, or predicting the optimal r and w_h could increase performance and robustness across different maps. Determining the reason behind WO-EECBS improved bound's negative performance effect would also be interesting investigative work.

Our experiments provide compelling evidence for MAPF practitioners to use Weighted EECBS and more broadly incorporate weighted cost-to-go heuristics. We first introduce WO-EECBS which incorporating the weighted cost-to-go in the open queue, and the analyze the effect of improving the lower bound on utilizing prioritized conflicts and symmetry reasoning. We then introduce WF-EECBS which modifies the focal priority to include a weighted cost-to-go and weighted conflict heuristic, discover a surprising relationship between the ratios of the weights, and show significant speeds up compared to EECBS. We demonstrate how these speeds ups change across different hyper-parameters w_{so}, w_h, r and different scenario (map sizes, numbers of agents). We show that our weighted focal technique results in similar speeds up the low level planner regardless of the high level search, illustrating how our technique is readily generalizable to other suboptimal CBS methods. Finally, we show that PP is actually just one specific step in suboptimal CBS with an infinite sub-optimality, and show W-EECBS is the natural generalization of the two.

Overall, our proposed methods bear no additional overhead and are directly usable in other CBS based suboptimal planners. More broadly, we hope this work inspires future MAPF work to incorporate other single agent path planner advancements (which usually rely in-part on weighted cost-to-go heuristics) into the MAPF domain.

Acknowledgement. This material is partially supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE1745016 and DGE2140739.

References

Andreychuk, A.; Yakovlev, K. S.; Atzmon, D.; and Stern, R. 2019. Multi-Agent Pathfinding (MAPF) with Continuous Time. *CoRR*, abs/1901.05506.

Barer, M.; Sharon, G.; Stern, R.; and Felner, A. 2014. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In *Seventh Annual Symposium on Combinatorial Search*.

Biswas, J.; Mendoza, J. P.; Zhu, D.; Choi, B.; Klee, S.; and Veloso, M. 2014. Opponent-driven planning and execution for pass, attack, and defense in a multi-robot soccer team. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 493–500.

Boyarski, E.; Felner, A.; Le Bodic, P.; Harabor, D. D.; Stuckey, P. J.; and Koenig, S. 2021. f-Aware Conflict Prioritization; Improved Heuristics For Conflict-Based Search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14): 12241–12248.

Boyarski, E.; Felner, A.; Stern, R.; Sharon, G.; Tolpin, D.; Betzalel, O.; and Shimony, E. 2015. ICBS: Improved conflict-based search algorithm for multi-agent pathfinding. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Čáp, M.; Novák, P.; Kleiner, A.; and Seleckỳ, M. 2015. Prioritized planning algorithms for trajectory coordination of multiple mobile robots. *IEEE transactions on automation science and engineering*, 12(3): 835–849.

Cohen, L.; Greco, M.; Ma, H.; Hernandez, C.; Felner, A.; Kumar, T. K. S.; and Koenig, S. 2018. Anytime Focal Search with Applications. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 1434–1441. International Joint Conferences on Artificial Intelligence Organization.

Erdmann, M.; and Lozano-Perez, T. 1987. On multiple moving objects. *Algorithmica*, 2(1): 477–521.

Holte, R. C.; Majadas, R.; Pozanco, A.; and Borrajo, D. 2019. Error Analysis and Correction for Weighted A*'s Suboptimality (Extended Version). *CoRR*, abs/1905.11346.

Kusnur, T.; Mukherjee, S.; Saxena, D. M.; Fukami, T.; Koyama, T.; Salzman, O.; and Likhachev, M. 2021. A planning framework for persistent, multi-UAV coverage with global deconfliction. In *Field and Service Robotics*, 459–474. Springer.

Li, J.; Chen, Z.; Harabor, D.; Stuckey, P. J.; and Koenig, S. 2022. MAPF-LNS2: Fast Repairing for Multi-Agent Path Finding via Large Neighborhood Search.

Li, J.; Felner, A.; Boyarski, E.; Ma, H.; and Koenig, S. 2019. Improved Heuristics for Multi-Agent Path Finding with Conflict-Based Search. In *IJCAI*, volume 2019, 442–449.

Li, J.; Gange, G.; Harabor, D.; Stuckey, P. J.; Ma, H.; and Koenig, S. 2020a. New techniques for pairwise symmetry breaking in multi-agent path finding. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, 193–201.

Li, J.; Harabor, D.; Stuckey, P. J.; and Koenig, S. 2021. Pairwise Symmetry Reasoning for Multi-Agent Path Finding Search. *CoRR*, abs/2103.07116.

Li, J.; Ruml, W.; and Koenig, S. 2021. Eecbs: A bounded-suboptimal search for multi-agent path finding. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 12353–12362.

Li, J.; Tinka, A.; Kiesel, S.; Durham, J. W.; Kumar, T. S.; and Koenig, S. 2020b. Lifelong Multi-Agent Path Finding in Large-Scale Warehouses. In *AAMAS*, 1898–1900.

Ma, H.; Harabor, D.; Stuckey, P. J.; Li, J.; and Koenig, S. 2019. Searching with consistent prioritization for multi-agent path finding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 7643–7650.

Phillips, M.; and Likhachev, M. 2011. SIPP: Safe interval path planning for dynamic environments. *2011 IEEE International Conference on Robotics and Automation*, 5628–5635.

Ratner, D.; and Warmuth, M. K. 1986. Finding a Shortest Solution for the N× N Extension of the 15-PUZZLE Is Intractable. In *AAAI*, 168–172.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Boyarski, E.; and Bartak, R. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. *Symposium on Combinatorial Search (SoCS)*, 151–158.

Sun, D.; and Mills, J. K. 2002. Adaptive synchronized control for coordination of multirobot assembly tasks. *IEEE Transactions on Robotics and Automation*, 18(4): 498–510.

Thayer, J. T.; and Ruml, W. ???? Faster Than Weighted A*: An Optimistic Approach to Bounded Suboptimal Search.

Thayer, J. T.; and Ruml, W. 2011. Bounded suboptimal search: A direct approach using inadmissible estimates. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

Velagapudi, P.; Sycara, K.; and Scerri, P. 2010. Decentralized prioritized planning in large multirobot teams. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4603–4609. IEEE.

Wu, W.; Bhattacharya, S.; and Prorok, A. 2020. Multi-robot path deconfliction through prioritization by path prospects. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 9809–9815. IEEE.

A Quick Recap

We provide a quick recap of our work for the skimming reader.

Recommended background reading

Before reading this paper, readers new to focal search are recommended to read (Cohen et al. 2018), and readers new to bounded sub-optimal CBS are recommended to read (Barer et al. 2014). Readers interested in our lower bound improvements and how they relate to bounded suboptimal CBS should read ECBS (Thayer and Ruml 2011) and EECBS (Li, Ruml, and Koenig 2021).

Intended takeaways

Main takeaways: We can effectively incorporate a weighted cost-go-heuristic in bounded suboptimal CBS (e.g. ECBS, EECBS) by modifying FOCAL to rather than modifying OPEN. Specifically, FOCAL should be changed from sorted by just c to sorted by $g+w_h*h+w_c*c$ where c is the number of conflicts on the current path. Additionally, the ratio of $r = w_c/w_h$ primarily determines performance rather than w_c or w_h, resulting in reparameterizing r and w_h with $f_{focal}(g, h, c) = g+w_h*(h+r*c)$. Our change can result in large (50+ speed ups) and solve significantly more problem instances. Our performance gain generalizes to both ECBS and EECBS, suggesting it can be directly helpful in other suboptimal CBS methods. Lastly, we show how (weighted) PP is a sub-step of (weighted) suboptimal CBS and how our method relates the two.

Weighted Open Variant (WO-EECBS)

We can incorporate a weighted cost-go-heuristic in the open list (OPEN) and keep the focal list (FOCAL) with the conflict heuristic un-changed. Doing so limits w_h by w_{so} as well as reduces the flexibility/effectiveness of FOCAL to maintain bounded sub-optimality. In experimental results, WO-EECBS does not produce much consistent speed-up.

Improving the lower bound increases utilization of prioritized conflicts and symmetry reasoning, but actually hurts runtime performance.

Weighted Focal Variant (WF-EECBS)

We keep OPEN unweighted and instead incorporate the weighted heuristic in FOCAL along with the inadmissible conflict heuristic. Changing the magnitude of w_h, w_c changes the relative importance of finding a solution fast (higher w_h) vs avoiding conflicts (higher w_c), allowing us to explicitly reason between the two. Our experiments show the performance is dominated by the ratio $r = w_c/w_h$ rather than the actual w_c or w_h weights, with optimal values $r \in [2, 16]$.

Overall WF-EECBS helps on 7 out of 8 maps, providing large speed ups (10+) on three and massive speed ups (50-100+) on two compared to regular EECBS. Weighted EECBS (W-EECBS) therefore refers to this weighted focal version. Weighting the cost-go-heuristic w_h helps more as the map size increases. Additionally, for each of the smaller maps, relative performance usually decreases as the number of agents increases. Both of these patterns fit our intuition; weighting the cost-to-go heuristic is more useful when paths are longer (larger maps) and less effective when there is more congestion (smaller maps with more agents) which would likely cause deviations from the heuristic.

Relating CBS, Prioritized Planning, and W-EECBS:

We prove that PP is actually equivalent to generating the initial agent paths in the root CT node in bounded sub-optimal CBS planners like (ECBS, EECBS) with an infinite sub-optimality and prove that W-EECBS is the naturalize generalization of weighted PP and EECBS. We shorthand our W-EECBS method with $r \leftarrow \infty$ as CBSPP, and experimentally verify how the number of generated nodes stays at 1 for low levels of agents until conflicts become unavoidable. We demonstrate how CBSPP's ability to replan using CBS's conflict resolution increases success rate compared to prioritized planning. When possible, practitioners using PP should instead use W-EECBS with a large sub-optimality as they get the same prioritized planning behavior in the root node along with the natural robustness and completeness of CBS.

B Justifying removing timeouts from plots

In all figures, if a method fails (times out on all 5 seeds) on a particular number of agents on a map, we do not report larger number of agents as this causes misleading visuals. A reminder that the speed up $S_{method} = T_{baseline}/T_{method}$ (larger is better) is reported to normalize differences in hardware, where the baseline is EECBS. Figure A1 demonstrates an example where including timeouts causes different methods to appear to have the same result, as well as causes false trends on their behaviour compared to the baseline.

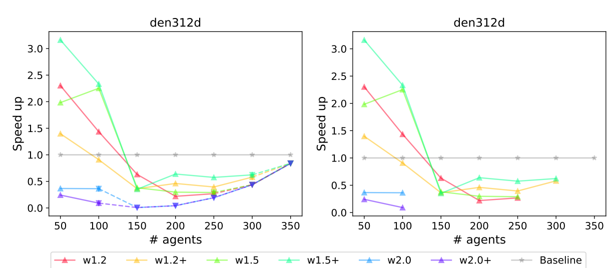

Figure A1: **Justifying removing timeouts —** The left image shows the raw speed up including instances which have timed out (in downward triangle marker and dashed lines) and instances which finished within the timeout T_{max} (in upward triangle marker, solid lines, and starting at the leftmost of each plot). We see that the timed out instances all have the same values at the bottom as the speed up $S_{method} = T_{baseline}/T_{method} = T_{baseline}/T_{max}$. This causes the false impression that different failed methods have the same speed up, and that the speed up increases as the number of agents increases (which is actually caused by $T_{baseline}$ increasing). The right map without these failed instances displays the results much more accurately.