Robust Multiagent Combinatorial Path Finding Technical Appendix

Anonymous submission

Proofs

In this section, we establish the theoretical guarantees of our approach. We begin by proving the correctness and completeness of the K-Best-Sequencing procedure used in *RCbssEff*, which consists of three steps: (1) reducing the problem of finding a goal sequence allocation γ (an mTSP instance) to an equivalent single-agent E-GTSP instance, (2) enumerating the K lowest-cost tours using the K-Best-EGTSP algorithm (Alg. 2), and (3) decomposing the K^{th} tour back into a multi-agent goal sequence allocation γ. Building on these results, we then prove that *RCbssEff*, based on the Robust CBSS framework (Alg. 1), is both complete and optimal.

Correctness of Step 1: Reducing the Problem of Finding γ (mTSP) to an Equivalent Single-Agent *E-GTSP* Instance

In Step 1, we reduce the problem of finding a goal sequence allocation γ in G (an mTSP instance) to an equivalent single-agent *E-GTSP* instance represented by G'. We prove that for any goal sequence allocation γ, there exists a corresponding *E-GTSP* tour with the same cost, ensuring that this reduction preserves both feasibility and cost.

Notation 1. *Let Tour(γ) denote the E-GTSP tour in G' constructed from a goal sequence allocation γ in G.*

Lemma 1. *For any feasible goal sequence allocation γ defined over G, there exists a corresponding E-GTSP tour Tour(γ) in G'.*

Proof. Let $\gamma = \langle sq(1), \dots, sq(n) \rangle$ be a feasible allocation in G, where $sq(i)$ is the ordered sequence of goals assigned to agent i.

Cluster construction. Each goal $g_j \in G$ is represented in G' as a cluster C_{g_j} containing one vertex per feasible approach orientation, ensuring each goal is represented exactly once with a chosen orientation. Each agent i is represented as a singleton cluster C_0^i containing its start position s_i, ensuring all agents are included.

Edge construction. Type-1 edges in G' connect agent starts to goals and goals to each other, with costs equal to the shortest-path traversal costs in G, preserving true movement costs. Zero-cost Type-2 edges connect goals to agent-start clusters and agent-start clusters to each other. These edges allow concatenating all agents' paths into a single E-GTSP tour and prevent enforcing a fixed destination goal.

Tour construction. We form Tour(γ) in G' by, for each agent i, starting at its singleton cluster C_0^i and visiting the orientation-specific vertex of its assigned goals in the order they appear in $sq(i)$, selecting exactly one vertex per cluster. After completing an agent's sequence, a Type-2 edge transitions to the next singleton cluster C_0^k, concatenating all sequences into a single E-GTSP tour.

Since every goal in γ appears exactly once across all agents and every agent's start position is included, the constructed path visits one vertex per cluster in G' and is thus a valid E-GTSP tour. Moreover, no fixed destination cluster is enforced, so the tour accurately reflects the original problem's lack of destination goal constraints. □

Lemma 2. *The cost of the E-GTSP tour Tour(γ) in G' equals the cost of the original allocation γ in G.*

Proof. The cost of each Type-1 edge between orientation-specific vertices (including edges from agent starts to goals) is inherited directly from the original cost function in G, which accounts for movement distances and orientations. Zero-cost Type-2 edges do not contribute to the cost. Thus, the total cost of Tour(γ) equals the total cost of executing the allocation γ in G. □

Theorem 3. *Let γ^* be an optimal goal sequence allocation in G. Then Tour(γ^*) is an optimal tour in G' for the corresponding E-GTSP instance.*

Proof. By Lemma 1, Step 1 constructs a valid E-GTSP tour in G' for any feasible allocation in G. By Lemma 2, this construction preserves the cost of the allocation. Therefore, the optimal allocation γ^* corresponds to a tour Tour(γ^*) with the same minimal cost in the E-GTSP formulation. □

Completeness of Step 2: Enumerating the K Lowest-Cost E-GTSP Tours (Alg. 2)

In Step 2, we enumerate the K lowest-cost E-GTSP tours in non-decreasing order of cost. We prove that the K-Best-EGTSP algorithm is complete, i.e., it returns all K lowest-cost tours whenever they exist.

Let $\mathcal{S}$ denote the set of all feasible tours for a given *E-GTSP* instance. A set of tours $\mathcal{S}_K \subseteq \mathcal{S}$ is a K*-lowest-cost* set if (1) $|\mathcal{S}_K| = K$, and (2) every tour $S' \in \mathcal{S} \setminus \mathcal{S}_K$ has cost at least as large as the most expensive tour in $\mathcal{S}_K$.

Theorem 4. *Given a complete and optimal E-GTSP solver, K-best-EGTSP is guaranteed to return a K-lowest-cost set of tours for any $K > 0$.*

Proof. Intuitively, *K-best-EGTSP* enumerates tours by systematically branching on edges of previously generated tours. Because the underlying *E-GTSP* solver is complete and optimal, each new branch eventually explores all feasible tours in non-decreasing order of cost.

Assume, for contradiction, that *K-best-EGTSP* fails to return a valid K-lowest-cost set. Then there exists a returned tour s'_j such that a cheaper tour s_j is missing, i.e., $\mathrm{cost}(s_j) < \mathrm{cost}(s'_j)$.

Case 1: The first call to `rEGTSP` is made with $I_c^0 = O_c^0 = \emptyset$, i.e., directly invoking the base *E-GTSP* solver. By assumption, the solver returns the optimal tour. If a cheaper tour s_j existed, it would have been selected over s'_j, a contradiction.

Case 2: Suppose s'_j is the K^{th} selected tour. If s_j was already in the priority queue when s'_j was chosen, the best-first ordering would require expanding s_j first, contradicting the selection of s'_j. If s_j was not yet generated when s'_j was selected, then it must eventually be produced in a later iteration, as *K-best-EGTSP* systematically explores all feasible tours by branching on previously generated ones with added inclusion/exclusion constraints. Crucially, `rEGTSP` solves these constrained subproblems by temporarily modifying edge costs: edges required by inclusion constraints receive a large negative offset to enforce their selection, while all other edges either retain their original costs or are penalized. After a tour is found, its cost is re-evaluated using the original edge costs, ensuring that no constrained subproblem can yield a tour with a strictly lower true cost than what could have been found in the unconstrained problem. Therefore, if s_j were truly cheaper than s'_j, it would necessarily have been discovered earlier during the enumeration, which contradicts the assumption.

Thus, no such s_j can exist, proving the algorithm's completeness. □

Correctness of Step 3: Decomposing the E-GTSP Tour into a Goal Sequence Allocation γ

In Step 3, we decompose an E-GTSP tour in G' into a goal sequence allocation γ in G. We prove that this decomposition preserves feasibility and cost, yielding an equivalent optimal allocation.

Notation 2. *Let Alloc$(tour)$ denote the goal sequence allocation in G obtained by decomposing a given E-GTSP tour in G'.*

Lemma 5. *For any E-GTSP tour in G', there exists a corresponding goal sequence allocation γ in G.*

Proof. Let $tour$ be an E-GTSP tour in G', represented as an ordered sequence of vertices $\{s_0, v_{g_{j,o}}, \ldots, s_i, \ldots, s_n, \ldots\}$, where each s_i is the initial configuration of agent i, and each $v_{g_{j,o}}$ is an orientation-specific copy of goal j approached from orientation o. Since each initial agent vertex forms a singleton cluster in G', the tour must include exactly n such agent vertices. To construct Alloc$(tour)$, we partition the tour into n disjoint segments by cutting at each agent vertex. For each agent i, let $sq(i)$ denote the sequence of goal vertices appearing between s_i and the next agent vertex. Each $sq(i)$ is assigned to agent i, preserving the visitation order. Finally, we map each orientation-specific goal vertex $v_{g_{j,o}} \in sq(i)$ in G' to its original goal v_{g_j} in G, discarding orientation information. This produces a valid goal sequence allocation Alloc$(tour)$ in G. □

Lemma 6. *The cost of the goal sequence allocation in G is equal to the cost of the original E-GTSP tour in G'.*

Proof. As described in Lemma 5, the allocation is constructed by partitioning the E-GTSP tour in G' into agent-specific segments. Segmentation occurs at Type-2 edges, which are auxiliary and have zero cost. Removing these edges does not affect the total cost. Additionally, mapping orientation-specific goal vertices $v_{g_{j,o}}$ in G' to their original vertices v_{g_j} in G does not alter the cost: the edge weights between goals in the allocation are inherited from the E-GTSP traversal, which already accounts for orientation-dependent movement. Thus, the total cost remains unchanged when mapping from G' back to G. □

Theorem 7. *Let tour be an optimal tour in G' for the E-GTSP. Then Alloc$(tour)$ is an optimal goal sequence allocation in G.*

Proof. By Lemma 5, Step 3 produces a valid allocation in G from any E-GTSP tour in G'. By Lemma 6, this decomposition preserves the total cost. Thus, if $tour$ is optimal for the E-GTSP in G', then Alloc$(tour)$ is also optimal for the corresponding goal sequence allocation problem in G. □

Completeness of *RCbssEff* (Based on the Robust CBSS Framework in Alg. 1)

In this section, we prove that *RCbssEff* is *complete*: if a feasible robust solution exists, the algorithm is guaranteed to return one.

Theorem 8. *Given the correctness and completeness of Steps 1–3 (which ensure the correctness and completeness of the K-Best-Sequencing procedure), RCbssEff is solution complete.*

Proof. Assume, for contradiction, that *RCbssEff* fails to return a solution for some solvable instance. Let π be a feasible robust solution, and let $\gamma(\pi)$ denote its corresponding goal sequence allocation.

RCbssEff maintains a set $\mathcal{T}$ of Constraint Trees (CTs), each representing a unique goal sequence allocation, and systematically explores them using CBS to find feasible solutions.

Case 1: *RCbssEff* never creates a CT tree for $\gamma(\pi)$. However, *RCbssEff* enumerates all goal sequence allocations using the K-Best-Sequencing method, which invokes *K-best-EGTSP*. Since *K-best-EGTSP* is complete, $\gamma(\pi)$ must eventually be generated, ensuring that a CT tree for it is created — a contradiction.

Case 2: A CT tree T for $\gamma(\pi)$ exists, but *RCbssEff* never finds π within it. Since π is a feasible robust solution consistent with $\gamma(\pi)$, it must correspond to some CT node in T. *RCbssEff* systematically expands all nodes in each CT until a robust solution is found or the tree is fully explored. Thus, π must eventually be reached and verified, contradicting the assumption.

Therefore, no feasible robust solution can be omitted, proving that *RCbssEff* is complete. □

Optimality of *RCbssEff* (Based on the Robust CBSS Framework in Alg. 1)

Next, we prove that the solution returned by *RCbssEff* is *optimal* with respect to the defined cost function.

Theorem 9. *Given the correctness and completeness of Steps 1–3 (which ensure the correctness and completeness of the K-Best-Sequencing procedure), the solution returned by RCbssEff is optimal.*

Proof. Let Z denote the CT node corresponding to the solution π returned by *RCbssEff*. By construction, π consists of tours where each agent visits all its assigned goals. As Z was expanded, every other CT node $Z' \in \texttt{OPEN}$ satisfied $\mathrm{cost}(Z') \geq \mathrm{cost}(Z)$.

Assume, for contradiction, that there exists another solution π' with $\mathrm{cost}(\pi') < \mathrm{cost}(\pi)$. Let $\gamma(\pi')$ denote the goal sequence allocation of π'.

Case 1: A CT tree for $\gamma(\pi')$ already exists. Then there must be an unexpanded node Z_T in this tree with $\mathrm{cost}(Z_T) \leq \mathrm{cost}(\pi')$. But since *RCbssEff* selected Z for expansion, we must have $\mathrm{cost}(Z) \leq \mathrm{cost}(Z_T)$, which contradicts $\mathrm{cost}(\pi') < \mathrm{cost}(\pi)$.

Case 2: No CT tree for $\gamma(\pi')$ has been generated. Then $\mathrm{cost}(Z) \leq \mathrm{cost}(\gamma(\pi'))$, where $\mathrm{cost}(\gamma(\pi'))$ is the minimal cost of its allocation. Since $\mathrm{cost}(\gamma(\pi')) \leq \mathrm{cost}(\pi')$, it follows that $\mathrm{cost}(\pi) \leq \mathrm{cost}(\pi')$, again contradicting the assumption.

Thus, no cheaper solution exists, and *RCbssEff* returns an optimal solution. □