**Distributed Conflict Based Search**

**Course: Collaboration in Artificial Intelligence**

**Problem: Path Finding**

**Project Guide: Prof. Roni Stern**

**By**

**Ran Weissman - 312236466**

**Uriel Zaed - 319435608**

**Content**

1. Multi Agent Path Finding…………………………………………..………….....3 - 4
2. Conflict Based Search……………………………..….………………….….......4 - 5
3. Distributed Conflict Based Search……………………………..…………….....5 - 7
4. Experiment……………..…………………………………………..………….......7 - 8
5. Results……………………………..……………….………………….…………...8 - 9
6. Conclusion………………………..……………….………………….…....................9
7. Future research……………………………..……………….……………...……9 - 10
8. Appendix…………………………………………………………………………11 - 12

# Multi Agent Path Finding

## Background

Multi-Agent Pathfinding (MAPF) is a complex problem that arises in various domains, such as robotics, transportation, and video games. It involves the task of determining collision-free paths for multiple agents moving in a shared environment while satisfying various constraints and objectives. These agents can be physical entities like robots or virtual entities like characters in a game. In recent years, there has been a significant surge in interest in MAPF due to its practical applications in autonomous systems and the need for efficient coordination among multiple agents.

## Problem Statement

The primary objective of MAPF is to find a set of collision-free paths for a group of agents so that they can reach their respective goals without colliding with each other or violating any constraints.

To achieve this objective, several key aspects of MAPF need to be addressed:

a) Path Planning: One of the critical challenges in MAPF is determining feasible paths for each agent to navigate from their initial positions to their respective goal locations while avoiding collisions with other agents and obstacles present in the environment.

b) Conflict Resolution: Conflicts can arise when agents' paths intersect or when there are conflicting goals or constraints. Resolving these conflicts is essential to ensure smooth and collision-free movement of the agents.

c) Optimality: Finding optimal or near-optimal solutions is another crucial aspect of MAPF. Optimization can involve minimizing various cost metrics such as total travel time, energy consumption, or the number of agents involved in conflicts. Obtaining such optimal or near-optimal solutions can greatly improve the efficiency and effectiveness of multi-agent systems.

d) Scalability: The ability to handle large-scale MAPF instances with numerous agents and complex environments is crucial. Developing efficient algorithms that can address scalability challenges is necessary to make MAPF practical and applicable in real-world scenarios.

## Existing Approaches

Various approaches have been proposed to tackle the MAPF problem. These approaches can be broadly classified into centralized and decentralized methods.

Centralized approaches formulate MAPF as a single optimization problem and aim to find a joint plan for all agents simultaneously. These methods often provide global optimality guarantees, but they can face challenges when it comes to scalability. As the number of agents or the complexity of the environment increases, the computational requirements of centralized approaches can suffer from scalability issues.

Decentralized approaches, on the other hand, decompose the MAPF problem into individual agent subproblems. Each agent plans its path independently while considering the presence of other agents. These methods scale well and can handle larger instances of MAPF efficiently. However, decentralized approaches may result in suboptimal solutions due to the lack of global coordination among agents.

# Conflict Based Search

## Background

Conflict-Based Search (CBS) is a widely used and effective algorithm for solving Multi-Agent Pathfinding (MAPF) problems. MAPF involves determining collision-free paths for multiple agents moving in a shared environment while satisfying various constraints and objectives. CBS addresses the challenges of coordinating multiple agents by employing conflict resolution techniques to find optimal or near-optimal solutions. CBS has gained significant attention in recent years due to its ability to handle large-scale MAPF instances with efficiency and scalability.

## CBS Approach

Conflict-Based Search adopts a two-phase process: the search phase and the conflict resolution phase.

a) Search Phase: In this phase, Conflict-Based Search (CBS) involves the utilization of a search algorithm, such as A\* (A-star) or Dijkstra's algorithm, to explore the space of possible paths for the agents. CBS treats each agent separately, enabling the search algorithm to generate a set of individual paths for each agent.

b) Conflict Resolution Phase: Once the search phase is complete, conflicts between agents are identified based on their intersecting paths or shared goals. CBS then applies conflict resolution techniques to modify the individual paths, ensuring that conflicts are resolved, and the agents can reach their goals without colliding. Common conflict resolution methods include prioritizing agents, swapping paths, or performing path precomputation.

## Key Features and Advantages of CBS

Conflict-Based Search offers several key features and advantages:

a) Optimality: CBS can find optimal solutions by iteratively resolving conflicts. Through the conflict resolution phase, the algorithm aims to minimize the total cost, such as travel time or energy consumption, or other relevant metrics.

b) Scalability: CBS is known for its scalability, allowing it to handle large-scale MAPF instances with numerous agents and complex environments. By decomposing the problem into individual agent subproblems, CBS can efficiently find solutions even in challenging scenarios.

c) Applicability: CBS has been successfully applied in various domains, including robotics, transportation, and game AI. Its versatility makes it suitable for a wide range of real-world applications that involve multi-agent coordination and path planning.

# Distributed Conflict Based Search

## Background

Distributed Conflict-Based Search (DCBS) is an extension of Conflict-Based Search (CBS) that addresses the challenges of multi-agent coordination in a distributed setting. Multi-Agent Pathfinding (MAPF) involves finding collision-free paths for multiple agents in a shared environment while satisfying various constraints. DCBS aims to distribute computational load and coordination efforts among agents, allowing for efficient and scalable solutions to MAPF problems.

## DCBS Approach

Search Phase: In this phase, similarly to normal CBS, a path is calculated for each agent. The main difference is that each agent calculates the path independently.

Conflict Resolution Phase: This phase in Distributed Conflict-Based Search (DCBS) aims to achieve a conflict-free solution, like CBS. However, in DCBS, conflicts are handled through communication and collaboration among agents. Each agent takes responsibility for resolving its individual conflicts by interacting and exchanging information with other agents.

## DCBS - pseudo code

Each agent plan individually - broadcast all

After getting all plans:

If Valid plans - broadcast all

else if agent has no conflicts - broadcast all

else agent has conflict

Constrain agent to avoid the conflict

and

broadcast to another agent to avoid the conflict

## Communication

Message passing is a crucial aspect of communication among agents in Distributed Conflict-Based Search (DCBS). Various types of messages are exchanged, each serving a specific purpose and conveying information.

## Path For Agent Message

During the initial step of communication in Distributed Conflict-Based Search (DCBS), agents send this type of message containing their respective paths. These messages are broadcast from each agent to all other agents involved in the system.

Handle receiving message, by taking the path from the message and put the path in the initial solution. If the agent receives all the paths, he creates root CBS node.

## Declare Solution Message

During the individual search, when a particular agent has reached a feasible solution, agents send message containing a feasible solution to the problem.

These messages are broadcast from each agent to all other agents involved in the system.

Handle receiving messages by checking if the message solution is better than his current solution. And update his current solution if necessary.

## Declare Conflict Message

During the individual search, when a certain agent finds a conflict that it is involved in with another agent, the first agent sends a message containing the CBS node and the conflict to the second agent.

Handle receiving messages by creating a new CBS node with a constraint according to the conflict.

## Declare Others Conflict Message

During the individual search, when a certain agent does not find a conflict in which it is involved, but the solution is not feasible, the agent sends a message containing CBS node.

These messages are broadcast from the agent to all other agents involved in the system.

Handle receiving messages, by checking a CBS node. If the agent has no conflict, he ignores it. If he has conflict, he creates a new CBS node with a constraint according to the conflict.

# Experiment

To evaluate the performance of DCBS, we implemented the algorithm in Python and utilized a diverse MAPF benchmark suite containing various problem instances. These benchmarks were intentionally designed to exhibit variations in size, agent count, and map complexity, facilitating a comprehensive assessment of DCBS's capabilities across different scenarios. As part of our analysis, we conducted a thorough comparison between DCBS and CBS algorithm, employing centralized method to provide a comprehensive evaluation.

The MAPF benchmark serves as a critical tool for evaluating and comparing the performance of various algorithms and approaches, allowing researchers and practitioners to develop more efficient and scalable solutions for real-world multi-agent coordination problems in domains such as robotics, logistics, and transportation.

Throughout the experimental process, we meticulously measured the efficiency of DCBS, focusing on key performance indicators such as runtime, the number of messages sent, and the Sum of cost. Additionally, we gauged the optimality of the solutions produced by DCBS by juxtaposing them against optimal solutions obtained through the centralized method, CBS. This evaluation provided valuable insights into the effectiveness of DCBS.

## The experiment parameters

Maps: random-32-32-10, empty-16-16

Scenarios: 1 (even)

Number of agents: from 2 to 50

The search time (runtime) has limited to 5 minutes for a single run.

# Results

The findings are presented in the appendix. There is a difference between the maps.

## Map random

Up to 36 agents, the DCBS demonstrates optimality. However, from 37 to 44 agents, the total cost of DCBS remains consistently lower than the optimal CBS by 2 to 3 steps.

The runtime of DCBS is slower compared to CBS, and a noticeable performance gap starts to emerge from 43 agents onward.

DCBS achieves success with up to 45 agents, mainly due to the limitation of the search time, which is set at 5 minutes.

## Map empty

Up to 20 agents, the DCBS demonstrates optimality. However, from 21 to 24 agents, the total cost of DCBS remains consistently higher than the optimal CBS by 1step.

The runtime of DCBS is slower compared to CBS.

DCBS achieves success with up to 24 agents, mainly due to the limitation of the search time, which is set at 5 minutes.

As the number of agents increases, the quantity of messages for both maps also experience becoming larger.

# Conclusion

DCBS algorithm is near optimal, the suboptimal part is sometimes lower and other time higher in the sum of cost compared to optimal CBS. Future research should focus on the optimality question and figure out if there is a problem in the A\* or other problem that causes it.

The runtime is slower than CBS and the success through 5 minutes limitation is partial.

The current state is that DCBS algorithm is suboptimal, and the run time is longer than CBS.

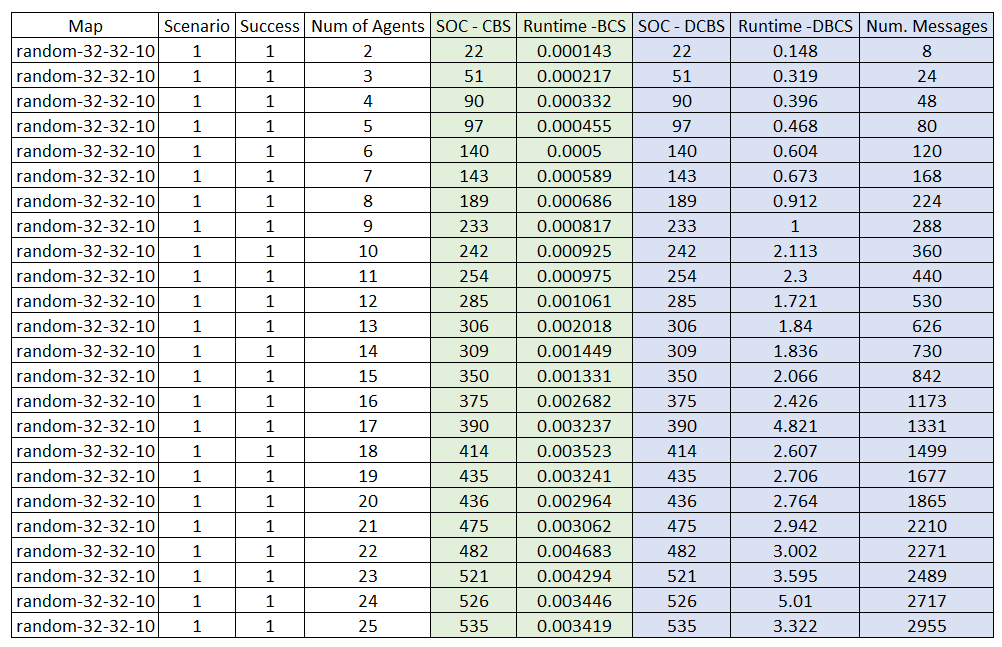
# Future research

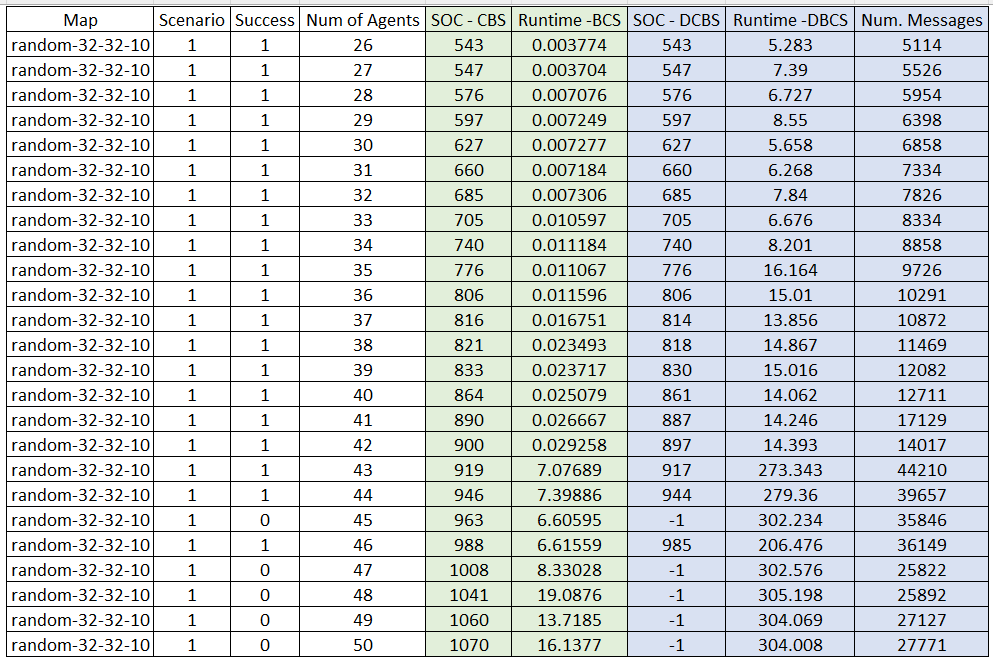
To advance our research in this domain, a comprehensive and rigorous investigation into the distinctions between DCBS and CBS is warranted. This examination will entail a thorough analysis of the A\* algorithm, followed by a logical comparison of the DCBS characteristics. By delving deeply into these aspects, our objective is to enhance our comprehension of the underlying mechanisms and identify any subtle variations between the two algorithms. Through this in-depth investigation, we aim to gain valuable insights into their respective strengths and weaknesses.

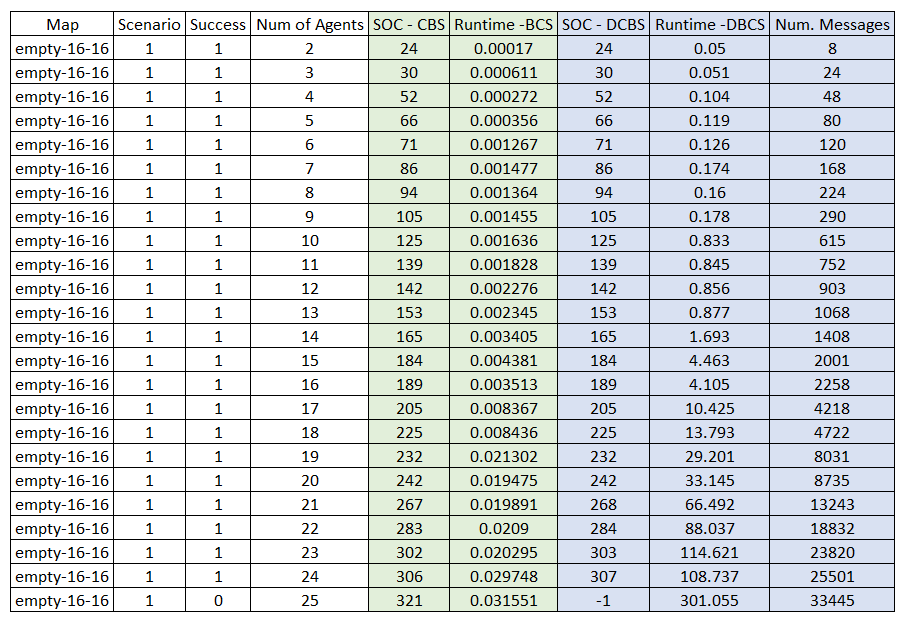
Moreover, we endeavor to devise a strategy for mitigating the generation of duplicate nodes resulting from the simultaneous initiation of searches by all agents from a shared node. A potential approach that merits further exploration involves designating a single agent to commence the search. This approach is very similar to the conventional CBS methodology. However, it is essential to subject this idea to extensive testing to ascertain its effectiveness. By implementing such a measure, we anticipate a substantial reduction in the volume of messages required to address the MAPF.

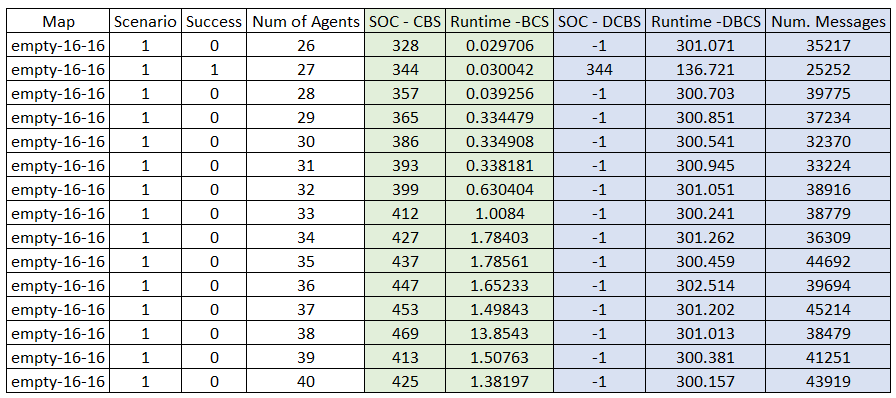
Lastly, the focus will be directed towards addressing the runtime aspect of the algorithm, primarily by adopting more sophisticated conflict selection strategies. Additionally, efforts will be made to enhance the efficiency of the A\* algorithm. Moreover, an array of other potential techniques shall be explored to further augment the algorithm's performance and capabilities.

# Appendix

1. **Results: map: random-32-32-10, scenario: 1, agents: 2 - 25**
2. **Results: map: random-32-32-10, scenario: 1, agents: 26 – 50**



1. **Results: map: empty-16-16, scenario: 1, agents: 2 - 25**

**4. Results: map: empty-16-16, scenario: 1, agents: 26 - 40**