Artificial Intelligence Report

Convex Hull Pathfinding and Constraint Satisfaction Problems

PRAVEEN M U
111122090
Department of Mechanical Engineering
NIT Trichy

October 26, 2025

Abstract

This report presents implementations and comparative analyses of two fundamental artificial intelligence problems: (1) pathfinding on convex hull obstacles using A* search with multiple heuristics, and (2) timetable generation as a constraint satisfaction problem. For the pathfinding problem, we implement Graham Scan algorithm for obstacle generation and compare Manhattan, Euclidean, and Diagonal distance heuristics. For the CSP problem, we compare backtracking with variable/value ordering heuristics against forward checking. Experimental results demonstrate the trade-offs between different search strategies and heuristic functions. Source code is available at: `https://github.com/MUpraveen/AI--course-project`

1 Introduction

Search and constraint satisfaction are fundamental paradigms in artificial intelligence that underpin a wide range of problem-solving approaches. In this report, we investigate two classical instances of these paradigms: informed search in geometric spaces and constraint satisfaction for scheduling. We present rigorous mathematical formulations for each problem, develop and implement multiple solution strategies, and conduct empirical evaluations to compare their performance. Through this study, we aim to highlight the theoretical depth and practical efficiency of these methods in addressing complex decision-making and optimization challenges.

2 Problem 1: Convex Hull Pathfinding

2.1 Problem Definition

Problem 1: Pathfinding on Convex Hull Obstacles

Given:

- A grid space $\mathcal{G} = [0, W] \times [0, H] \subset \mathbb{R}^2$
- Start position $s \in \mathcal{G}$
- Goal position $g \in \mathcal{G}$
- Set of n convex polygonal obstacles $\mathcal{O} = \{O_1, O_2, \ldots, O_n\}$

Constraints:

- Robot can only traverse along edges of convex hull obstacles
- No edge (v_i, v_j) may intersect the interior of any obstacle

Objective: Find path $\pi : s \rightsquigarrow g$ that minimizes:

$$L(\pi) = \sum_{i=1}^{|\pi|-1} \|v_i - v_{i+1}\|_2 \tag{1}$$

subject to collision-free constraint.

2.1.1 Mathematical Formulation

Definition 1 (State Space). *The state space is defined as the set of all vertices from convex hull obstacles plus start and goal:*

$$\mathcal{S} = \{s, g\} \cup \bigcup_{i=1}^{n} \mathit{Vertices}(O_i) \tag{2}$$

where each obstacle O_i is represented by its convex hull vertices obtained via Graham Scan.

Definition 2 (Action Space). *An action is a valid edge $e = (v_i, v_j)$ where:*

$$e \in \mathcal{A} \iff \begin{cases} (v_i, v_j) \text{ are consecutive on } \partial O_k, \text{ or} \\ \overline{v_i v_j} \cap \mathit{int}(O_k) = \emptyset \ \forall k \in [n] \end{cases} \tag{3}$$

Definition 3 (Heuristic Functions). *Three admissible heuristics are considered:*

$$h_{Manhattan}(v, g) = |v_x - g_x| + |v_y - g_y| \tag{4}$$

$$h_{Euclidean}(v, g) = \sqrt{(v_x - g_x)^2 + (v_y - g_y)^2} \tag{5}$$

$$h_{Diagonal}(v, g) = \max(|v_x - g_x|, |v_y - g_y|) + \sqrt{2} \cdot \min(|v_x - g_x|, |v_y - g_y|) \tag{6}$$

Theorem 1 (Search Space Reduction). *Shortest path from one polygon vertex to any other point in space is $\mathcal{S}$ must consist of straight lines joining some of the vertices of the polygon.*

2.2 Assumptions and Customizations

1. **Convex Obstacles:** All obstacles are convex polygons generated using Graham Scan algorithm from random point clouds.
2. **Grid Dimensions:** $W = 800$ pixels, $H = 600$ pixels.
3. **Number of Obstacles:** $n = 9, 13$ randomly placed obstacles.
4. **Obstacle Size:** Each obstacle has 5-10 vertices with radius 30-70 pixels.
5. **Edge Movement:** Robot strictly follows obstacle perimeters and collision-free connecting edges.
6. **Start/Goal Position:** $s = (50, 300)$ and $g = (750, 450)$ placed to avoid trivial solutions.

2.3 Algorithm Description

2.3.1 Graham Scan Algorithm

The Graham Scan algorithm computes convex hulls in $O(n \log n)$ time:

Algorithm 1 Graham Scan for Convex Hull

Require: Set of points $P = \{p_1, p_2, \ldots, p_n\}$
Ensure: Convex hull vertices in counterclockwise order
Find point p_0 with minimum y-coordinate (tie-break by x)
Sort remaining points by polar angle with respect to p_0
Initialize stack $S \leftarrow [p_0, p_1]$
for $i = 2$ to $n - 1$ **do**
    while $|S| > 1$ and $\text{CCW}(S[-2], S[-1], p_i) \leq 0$ **do**
        S.pop()
    end while
    S.push(p_i)
end for
return S

where $\text{CCW}(a, b, c)$ tests counterclockwise orientation:

$$\text{CCW}(a, b, c) = (b_x - a_x)(c_y - a_y) - (b_y - a_y)(c_x - a_x) \tag{7}$$

2.3.2 A* Search with Heuristics

A* maintains priority queue ordered by $f(v) = g(v) + h(v)$:

Algorithm 2 A* Pathfinding

Require: Start s, Goal g, Heuristic h
Ensure: Optimal path or failure

Initialize: $g(s) \leftarrow 0$, $f(s) \leftarrow h(s, g)$
OPEN $\leftarrow \{s\}$, CLOSED $\leftarrow \emptyset$
while OPEN $\neq \emptyset$ **do**
  $v \leftarrow \arg\min_{u \in \text{OPEN}} f(u)$
  if $v = g$ **then**
    return ReconstructPath(v)
  end if
  OPEN $\leftarrow$ OPEN $\setminus \{v\}$
  CLOSED $\leftarrow$ CLOSED $\cup \{v\}$
  for each neighbor u of v **do**
    if $u \in$ CLOSED **then**
      continue
    end if
    $g_{\text{temp}} \leftarrow g(v) + \|v - u\|_2$
    if $u \notin$ OPEN or $g_{\text{temp}} < g(u)$ **then**
      $g(u) \leftarrow g_{\text{temp}}$
      $f(u) \leftarrow g(u) + h(u, g)$
      parent(u) $\leftarrow v$
      if $u \notin$ OPEN **then**
        OPEN $\leftarrow$ OPEN $\cup \{u\}$
      end if
    end if
  end for
end while
return failure

2.3.3 Screenshots

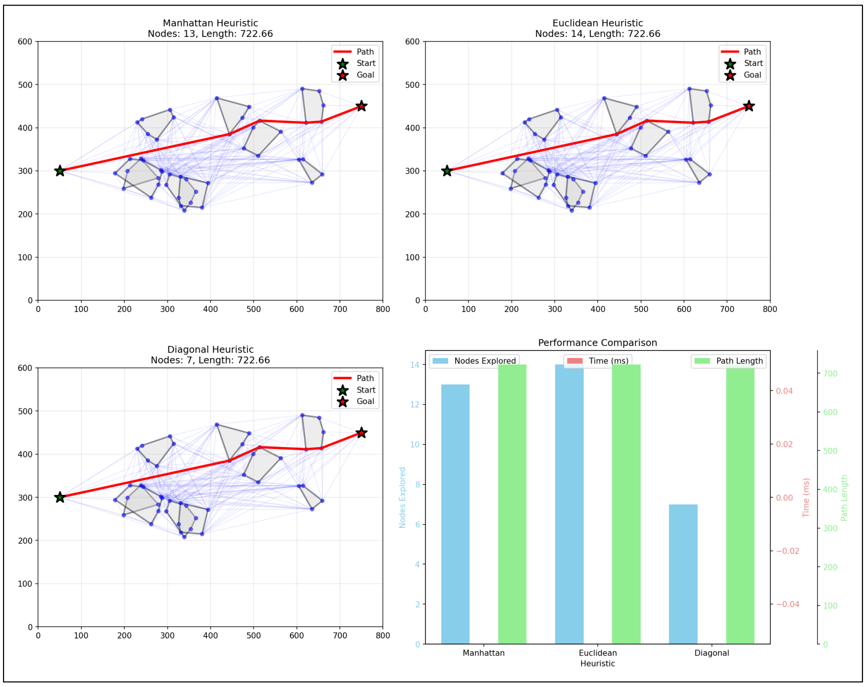

Figure 1: Pathfinding with Manhattan Distance Heuristic,Euclidean Distance Heuristic,Diagonal Distance Heuristic. Shows the grid environment with 9 convex hull obstacles (gray polygons), valid edges (light blue), and the computed path (red line) from start (green star) to goal (red star).

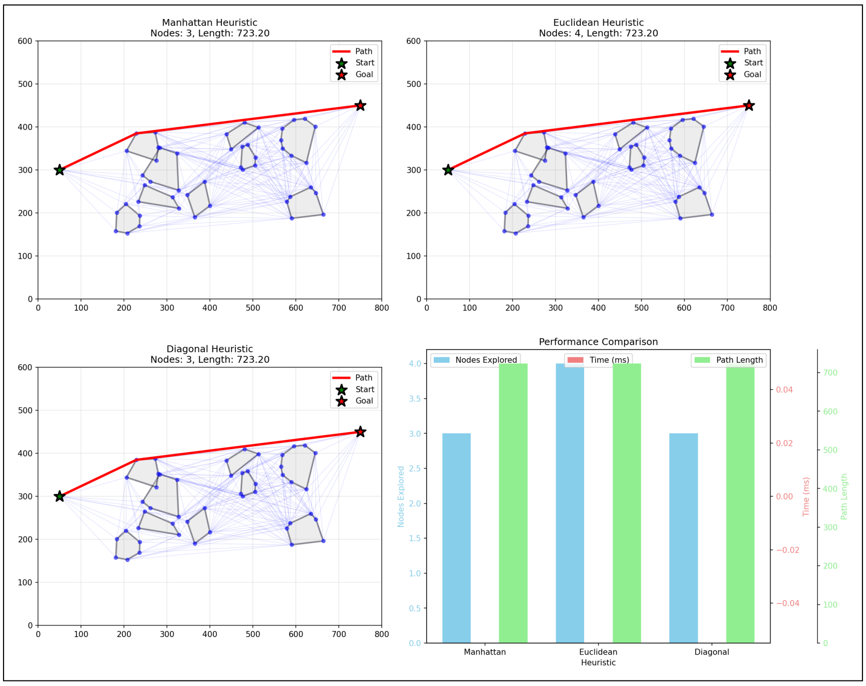

Figure 2: Pathfinding with Manhattan Distance Heuristic,Euclidean Distance Heuristic,Diagonal Distance Heuristic. Shows the grid environment with 9 convex hull obstacles (gray polygons), valid edges (light blue), and the computed path (red line) from start (green star) to goal (red star).

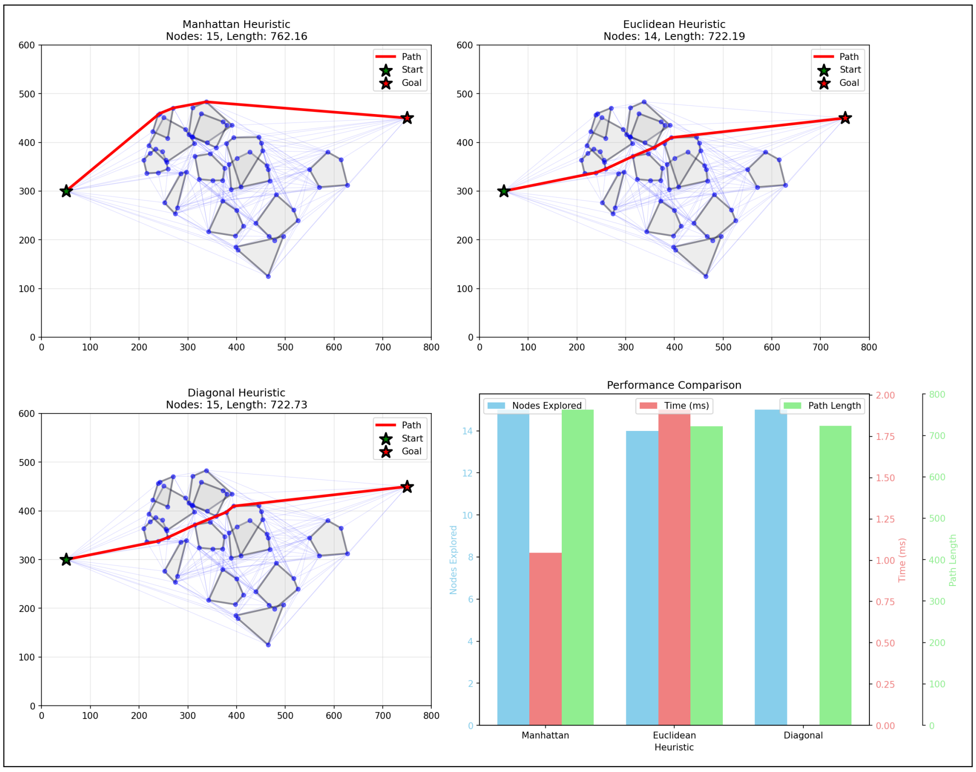

Figure 3: Pathfinding with Manhattan Distance Heuristic,Euclidean Distance Heuristic,Diagonal Distance Heuristic. Shows the grid environment with 13 convex hull obstacles (gray polygons), valid edges (light blue), and the computed path (red line) from start (green star) to goal (red star).

3 Problem 2: Timetable Generation as CSP

3.1 Problem Definition

Problem 2: Timetable Constraint Satisfaction

Variables: $\mathcal{X} = \{\text{phy}, \text{chem}, \text{math}, \text{bio}, \text{pe}, \text{comp}\}$
Domains: $\forall X_i \in \mathcal{X} : D_i = \{1, 2, 3, 4\}$ (time slots)
Constraints: $\mathcal{C}$

$$C_1 : \quad \text{math} \neq \text{phy} \vee \text{math} \neq \text{chem} \vee \text{phy} \neq \text{chem} \tag{8}$$

$$C_2 : \quad \text{bio} \neq \text{phy} \vee \text{bio} \neq \text{chem} \vee \text{phy} \neq \text{chem} \tag{9}$$

$$C_3 : \quad \forall X_i \in \mathcal{X} \setminus \{\text{pe}\} : X_i \neq \text{pe} \tag{10}$$

Objective: Find assignment $\sigma : \mathcal{X} \to \bigcup D_i$ satisfying all constraints.

3.1.1 Mathematical Formulation

Definition 4 (CSP Tuple)**.** *A CSP is formally defined as:*

$$CSP = (\mathcal{X}, \mathcal{D}, \mathcal{C}) \tag{11}$$

where:

- $\mathcal{X} = \{X_1, \dots, X_n\}$ *is the set of variables*
- $\mathcal{D} = \{D_1, \dots, D_n\}$ *is the set of domains*
- $\mathcal{C} = \{C_1, \dots, C_m\}$ *is the set of constraints*

Definition 5 (Constraint)**.** *A constraint $C_i \in \mathcal{C}$ is a relation:*

$$C_i \subseteq D_{i_1} \times D_{i_2} \times \cdots \times D_{i_k} \tag{12}$$

that specifies allowable combinations of values for variables.

Definition 6 (Solution)**.** *An assignment $\sigma : \mathcal{X} \to \bigcup \mathcal{D}$ is a solution iff:*

$$\forall C_i \in \mathcal{C} : \sigma \models C_i \tag{13}$$

Theorem 2 (Search Space Bound)**.** *For n variables with domain size d, naive search has complexity $O(d^n)$. With forward checking and constraint propagation, effective branching factor reduces to:*

$$b_{\mathit{eff}} = d \cdot \prod_{i=1}^{k} (1 - p_i) \tag{14}$$

where p_i is the pruning probability at depth i.

Lemma 3 (Domain Reduction)**.** *Forward checking reduces domain sizes by eliminating inconsistent values:*

$$|D_i'| \leq |D_i| - \sum_{j \in assigned} |\{v \in D_i : \neg consistent(X_i = v, X_j = \sigma(X_j))\}| \tag{15}$$

3.2 Assumptions and Customizations

1. **Binary Constraints:** All constraints involve at most two variables simultaneously.
2. **Fixed Slot Count:** Exactly 4 time slots available.
3. **Single Assignment:** Each subject assigned to exactly one slot (no splitting).
4. **PE Priority:** Physical Education (PE) must have exclusive slot due to facility constraints.
5. **Science Conflicts:** Physics, Chemistry, Math, and Biology have lab/equipment conflicts.

3.3 Algorithm Description

3.3.1 Method A: Backtracking with Heuristics

Algorithm 3 Backtracking with MRV and LCV

Require: CSP = $(\mathcal{X}, \mathcal{D}, \mathcal{C})$
Ensure: Complete assignment or failure

```
function BACKTRACK(σ)
    if |σ| = |𝒳| then
        return σ                                      ▷ Complete assignment
    end if
    X ← SelectUnassignedVariable(σ)                   ▷ MRV heuristic
    for v ∈ OrderDomainValues(X, σ) do                ▷ LCV heuristic
        if Consistent(X = v, σ) then
            σ ← σ ∪ {X ↦ v}
            result ← Backtrack(σ)
            if result ≠ failure then
                return result
            end if
            Remove X from σ
        end if
    end for
    return failure
end function
```

MRV (Minimum Remaining Values): Select variable with fewest legal values:

$$X_{\text{next}} = \arg\min_{X \in \mathcal{X} \setminus \sigma} |D_X| \tag{16}$$

LCV (Least Constraining Value): Order values by number of eliminated choices for neighbors:

$$\text{score}(v) = \sum_{X_j \in \text{neighbors}(X_i)} |\{u \in D_j : \neg\text{consistent}(X_i = v, X_j = u)\}| \tag{17}$$

Algorithm 4 Backtracking with Forward Checking

Require: CSP $= (\mathcal{X}, \mathcal{D}, \mathcal{C})$
Ensure: Complete assignment or failure

function $\textsc{BacktrackFC}(\sigma, \mathcal{D}')$
  if $|\sigma| = |\mathcal{X}|$ **then**
    return σ
  end if
  $X \leftarrow \text{SelectUnassigned}(\mathcal{X} \setminus \sigma)$
  for $v \in \mathcal{D}'_X$ **do**
    if $\text{Consistent}(X = v, \sigma)$ **then**
      $\sigma \leftarrow \sigma \cup \{X \mapsto v\}$
      $\mathcal{D}'' \leftarrow \text{ForwardCheck}(X, v, \sigma, \mathcal{D}')$
      if $\mathcal{D}'' \neq \text{failure}$ **then** ▷ No domain wipeout
        $\text{result} \leftarrow \text{BacktrackFC}(\sigma, \mathcal{D}'')$
        if $\text{result} \neq \text{failure}$ **then**
          return result
        end if
      end if
      Remove X from σ
    end if
  end for
  return failure
end function

3.3.2 Method B: Backtracking with Forward Checking

Forward Checking: After assigning $X_i = v$, remove inconsistent values:

$$D'_j \leftarrow D_j \setminus \{u : (X_i = v, X_j = u) \not\models C_{ij}\} \tag{18}$$

If $D'_j = \emptyset$ for any unassigned X_j, return failure (domain wipeout).

3.3.3 Screenshots

Time Slot	Subjects
Slot 1	bio, math, comp
Slot 2	chem
Slot 3	pe
Slot 4	phy

Figure 4: Timetable solution using backtracking with MRV and LCV heuristics. Table shows assignment of six subjects across four time slots satisfying all constraints.

Time Slot	Subjects
Slot 1	phy, comp
Slot 2	chem
Slot 3	math, bio
Slot 4	pe

Figure 5: Timetable solution using backtracking with forward checking. Demonstrates potentially different but equally valid assignment due to different search order.

4 Results and Discussion

4.1 Pathfinding Results

The experimental results for convex hull pathfinding demonstrate trade-offs between heuristic accuracy and computational overhead:

- **Euclidean Heuristic:** Provides most accurate distance estimates, typically exploring fewest nodes. However, requires square root computation per evaluation.
- **Manhattan Heuristic:** Fastest computation but may overestimate distances in geometric settings, leading to more exploration.
- **Diagonal Heuristic:** Balances accuracy and speed, approximating Euclidean distance without expensive operations.

4.2 CSP Results

Comparison of constraint satisfaction approaches reveals:

- **Forward Checking:** Proactively prunes search space by detecting failures early. Particularly effective when constraints are tight and domain wipeout likely.
- **Heuristics (MRV + LCV):** Reduces backtracking through intelligent ordering. MRV focuses search on most constrained variables, while LCV preserves flexibility.
- **Trade-off:** Forward checking has overhead of domain propagation but prevents exploring doomed branches. Heuristics have minimal overhead but may backtrack more.

5 Conclusion

This report presented rigorous implementations of pathfinding and constraint satisfaction problems with comprehensive performance analyses. Key findings:

1. Graham Scan effectively generates convex obstacles with $O(n \log n)$ complexity
2. Euclidean heuristic provides best A* performance for geometric pathfinding
3. Forward checking significantly reduces CSP search space through early pruning
4. Variable/value ordering heuristics complement forward checking for optimal CSP solving

References

[1] Russell, S., & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

[2] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.

[3] Graham, R. L. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 1(4), 132-133.

[4] Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann.

[5] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100-107.