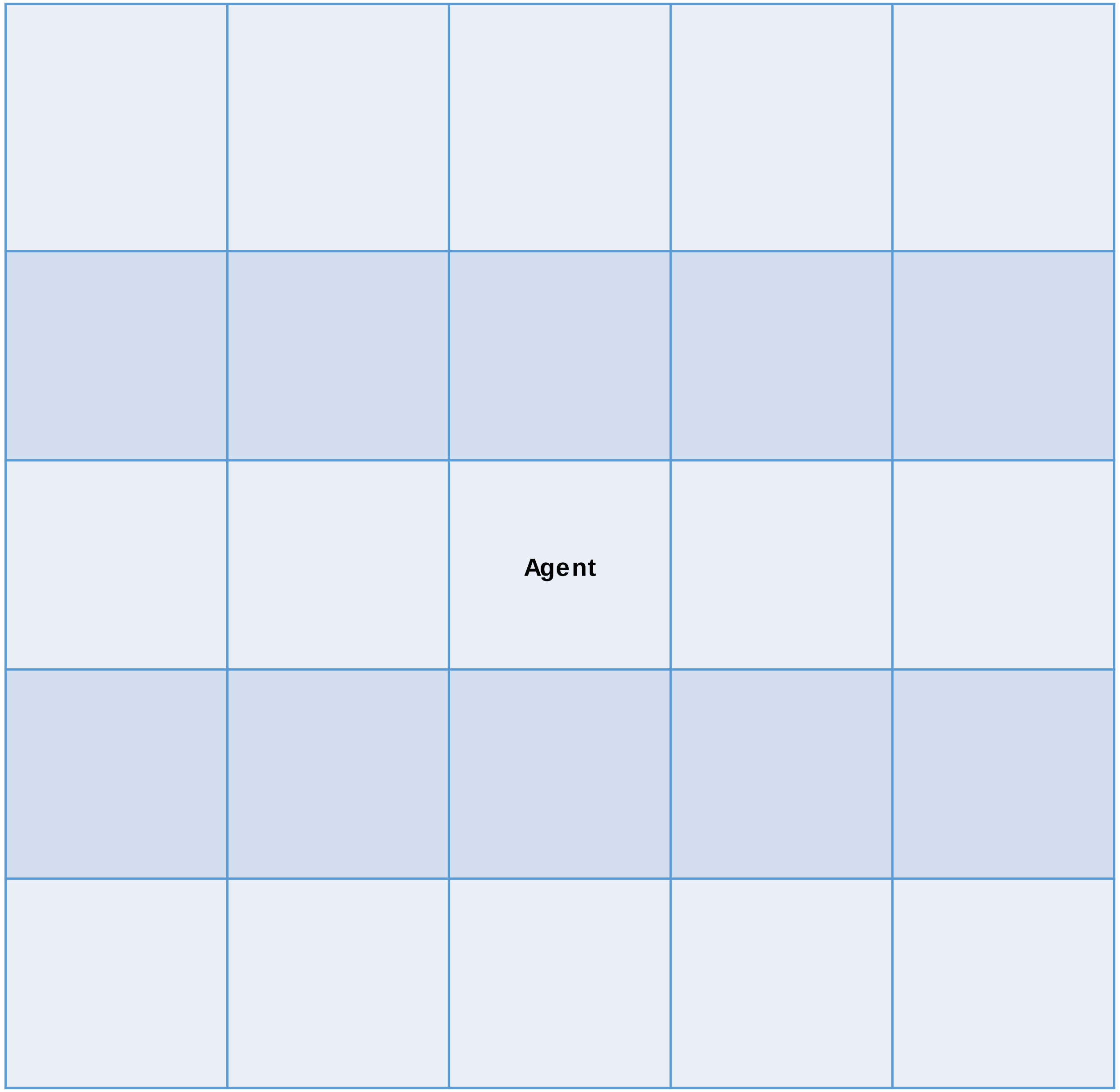
COMP90054 Workshop 3

Geye Guo

Recap: Manhattan Problem

- Consider an $m \times m$ **Manhattan Grid**, and a set of coordinates G to visit in any order.
- **Hint**: Consider a set of coordinates V' remaining to be visited, or a set of coordinates V already visited. What's the difference between them
- Formulate a state-based search problem to find a tour of all the desired points (i.e. define a state space, applicable actions, transition and cost functions).

m x m Grid



P = {S, s0, SG, A, T, C}

- a set of coordinates G to visit in any order
- Using a set of coordinates V' remaining to be visited,

Recap: Manhattan Problem

- a set of coordinates G to visit in any order
- Using a set of coordinates V' remaining to be visited

- $S = \{\langle x, y, V' \rangle \mid x, y \in \{0, \ldots, m-1\} \wedge V' \subseteq G\}$
- $s_0 = \langle (0,0), G \setminus \{(0,0)\} \rangle$
- $S_G = \{\langle (x,y), \{\} \rangle \mid x, y \in \{0, \ldots, m-1\}\}$
- $A(\langle x, y, V' \rangle) = \{(dx, dy) \mid$
  - $dx, dy \in \{-1, 0, 1\}$
  - $\wedge\ |dx| + |dy| = 1$
  - $\wedge\ x + dx, y + dy \in \{0, \ldots, m-1\}$
  - $(x + dx, y + dy) \notin W\ \}$
- $T(\langle x, y, V' \rangle, (dx, dy)) = \langle x + dx, y + dy, V' \setminus \{(x + dx, y + dy)\} \rangle$
- $c(a, s) = 1$

Problem 1

Reformulate the state-space model from *Review and Recap* as a STRIPS problem $P = \langle F, O, I, G \rangle$

STRIPS Model

- A **problem** in STRIPS is a tuple $P = \langle F, O, I, G \rangle$:
  - F stands for set of all atoms (boolean vars)
  - O stands for set of all operators (actions)
  - $I \subseteq F$ stands for initial situation
  - $G \subseteq F$ stands for goal situation

STRIPS Model

- Operators $o \in O$ **represented** by
  - the Add list $Add(o) \subseteq F$
  - the Delete list $Del(o) \subseteq F$
  - the Precondition list $Pre(o) \subseteq F$

Recap: Manhattan Problem

- a set of coordinates G to visit in any order
- Using a set of coordinates V' remaining to be visited,

- $S = \{\langle x, y, V' \rangle \mid x, y \in \{0, \ldots, m-1\} \land V' \subseteq G\}$
- $s_0 = \langle (0,0), G \setminus \{(0,0)\} \rangle$
- $S_G = \{\langle (x, y), \{\} \rangle \mid x, y \in \{0, \ldots, m-1\}\}$
- $A(\langle x, y, V' \rangle) = \{(dx, dy) \mid$
  - $dx, dy \in \{-1, 0, 1\}$
  - $\land\ |dx| + |dy| = 1$
  - $\land\ x + dx, y + dy \in \{0, \ldots, m-1\}$
  - $(x + dx, y + dy) \notin W\ \}$
- $T(\langle x, y, V' \rangle, (dx, dy)) = \langle x + dx, y + dy, V' \setminus \{(x + dx, y + dy)\} \rangle$
- $c(a, s) = 1$

Problem 1

- $F = \{at(x,y), visited(x,y) \mid x, y \in \{0, \ldots, m-1\}\}$
- $O = \{move(x,y,x',y')$:
  - Prec:
  - Add:
  - Del:
- $I = \{at(0,0), visited(0,0)\}$
- $G = \{visited(x,y) \mid (x,y) \in G'\}$

Problem 1

- $F = \{at(x, y), visited(x, y) \mid x, y \in \{0, \ldots, m-1\}\}$
- $O = \{move(x, y, x', y')$:
  - Prec: $at(x, y)$
  - Add: $at(x', y'), visited(x', y')$
  - Del: $at(x, y)$

  $\mid$ for each adjacent $(x, y), (x', y')$, and $(x', y') \notin W$ $\}$
- $I = \{at(0, 0), visited(0, 0)\}$
- $G = \{visited(x, y) \mid (x, y) \in G'\}$

Problem 1

notWall(x,y)

- $F = \{at(x, y), visited(x, y) \mid x, y \in \{0, \ldots, m-1\}\}$
- $O = \{move(x, y, x', y')$:
  - Prec: $at(x, y)$ notWall(x', y')
  - Add: $at(x', y'), visited(x', y')$
  - Del: $at(x, y)$

  $\mid$ for each adjacent $(x, y), (x', y')$
- $I = \{at(0, 0), visited(0, 0)\}$
- $G = \{visited(x, y) \mid (x, y) \in G'\}$

Problem 2

G1						G2
	Wall	**Wall**	**Starting point**			

Problem 2

- Goal-counting
- Manhattan distance to the closest goal(position) heuristic
- Manhattan distance to the furthest goal(position) heuristic

- Your own heuristic…

Recap: Dominant Relation

- If heuristic h1 dominates heuristic h2:
- Then we will have h1(s) >= h2(s), for all s belongs to state space S
- And both h1 and h2 need to be admissible

Recap: Dominant Relation

- If heuristic h1 dominates heuristic h2:
- Then we will have h1(s) >= h2(s), for all s belongs to state space S
- And both h1 and h2 need to be admissible

Common Heuristic for Manhattan Problems

- Null Heuristic: h = 0 for all state
- Goal counting: h = |V'|
- Manhattan to the closest: h = min(Manhattan(all food))
- Manhattan to the furthest: h = max(Manhattan(all food))
- Average of these two: h = average(min, max)
- Sum Manhattan: h = sum(Manhattan(all food))
- Minimum Spanning Tree: h = len(minimum spanning tree(all food))

Properties of Heuristic functions

Definition (Safe/Goal-Aware/Admissible/Consistent). *Let* Π *be a planning task with state space* $\Theta_\Pi = (S, L, c, T, I, S^G)$*, and let* h *be a heuristic for* Π*. The heuristic is called:*

- *safe if* $h^*(s) = \infty$ *for all* $s \in S$ *with* $h(s) = \infty$*;*
- *goal-aware if* $h(s) = 0$ *for all goal states* $s \in S^G$*;*
- *admissible if* $h(s) \leq h^*(s)$ *for all* $s \in s$ *;*
- *consistent if* $h(s) \leq h(s') + c(a)$ *for all transitions* $s \xrightarrow{a} s'$*.*

Is Manhattan furthest heuristic consistent?

- Being consistent: reduced value h(s) - h(s') <= 1

- Every time perform an action using this heuristic,
- There are 3 possible outcome:

1. one step closer: h(s) - h(s') = 1, also consistent
2. one step further: h(s) - h(s') = -1, also consistent
3. furthest food change

Furthest Manhattan heuristic

	G1		Agent			G2
	Wall	Wall				

Furthest Manhattan heuristic

- For state s, agent position: P1, furthest coordinate: G1
- For state s', agent position: P2, furthest coordinate: G2

\- h(s) = Distance(P1, G1), h(s') = Distance(P2, G2)

And for being consistent, we need to prove **h(s) <= h(s') + 1**

If h(s) > h(s') + 1, then it is not consistent

However, is it even possible that h(s) > h(s')?

Furthest Manhattan heuristic

For state s, agent position: P1, furthest coordinate: G1
For state s', agent position: P2, furthest coordinate: G2
- h(s) = Distance(P1, G1), h(s') = Distance(P2, G2)

Is it possible that h(s) > h(s')?
- Which means h(s) >= h(s') + 1, i.e, h(s') <= h(s) - 1

And we also have
- h(s) - 1 <= Distance(P2, G1) <= h(s) + 1
- as Distance(P1, P2) == 1

In conclusion,
- h(s') = Distance(P2, G2) <= Distance(P2, G1)
- **Furthest food should not change (which conflict with our assumption)**

Sum Manhattan heuristic

			Agent	G1	G2	
	Wall	Wall				

Minimum Spanning Tree Heuristic