ON SAT-BASED APPROACHES FOR MULTI-AGENT PATH FINDING WITH THE SUM-OF-COSTS OBJECTIVE

MATOUŠ MAŘÍK

SEMINAR OF ARTIFICIAL INTELLIGENCE II

PRESENTATION STRUCTURE

- Introduction
  - Problem Definition
    - Makespan Optimal Model
      - Sum of Costs Optimal Model
        - Model 1
          - Model 2
            - Reduction of Used Variables
              - Experiments

INTRODUCTION

MAPF – multi-agent path finding

INTRODUCTION

MAPF – multi-agent path finding

- real life motivation

INTRODUCTION

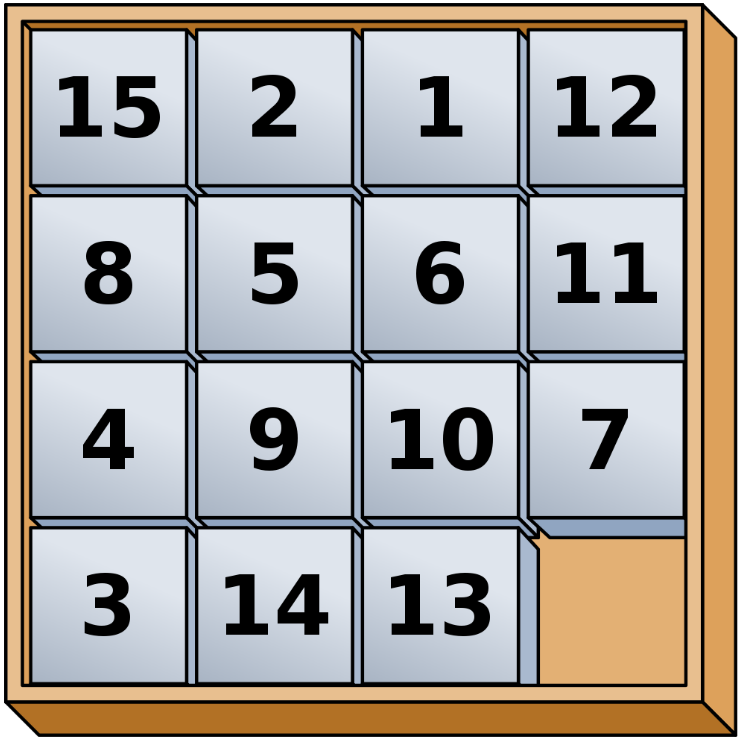
MAPF – multi-agent path finding

- real life motivation
- environment abstraction – graph
  (with constant distances)

INTRODUCTION

MAPF – multi-agent path finding

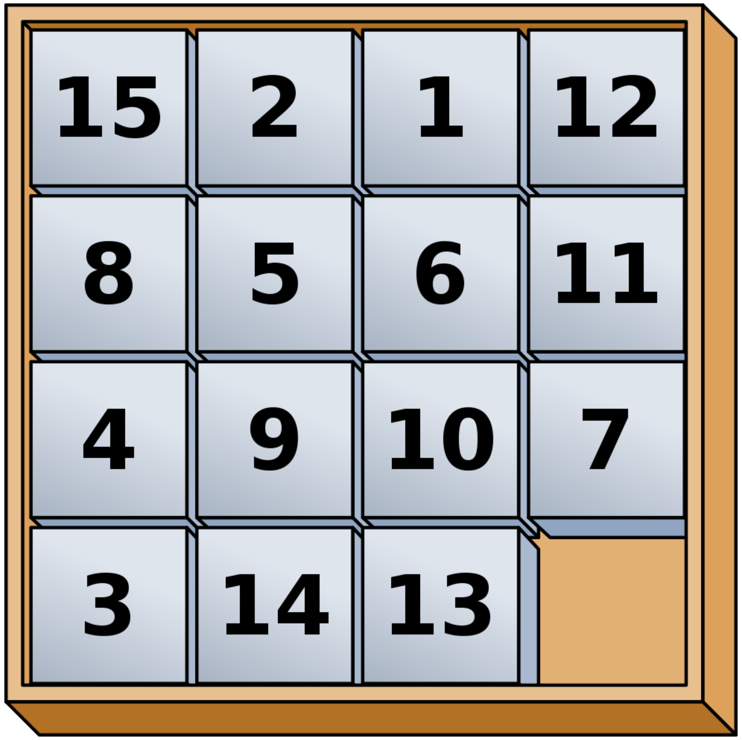
- real life motivation
- environment abstraction – graph (with constant distances)



INTRODUCTION

MAPF – multi-agent path finding

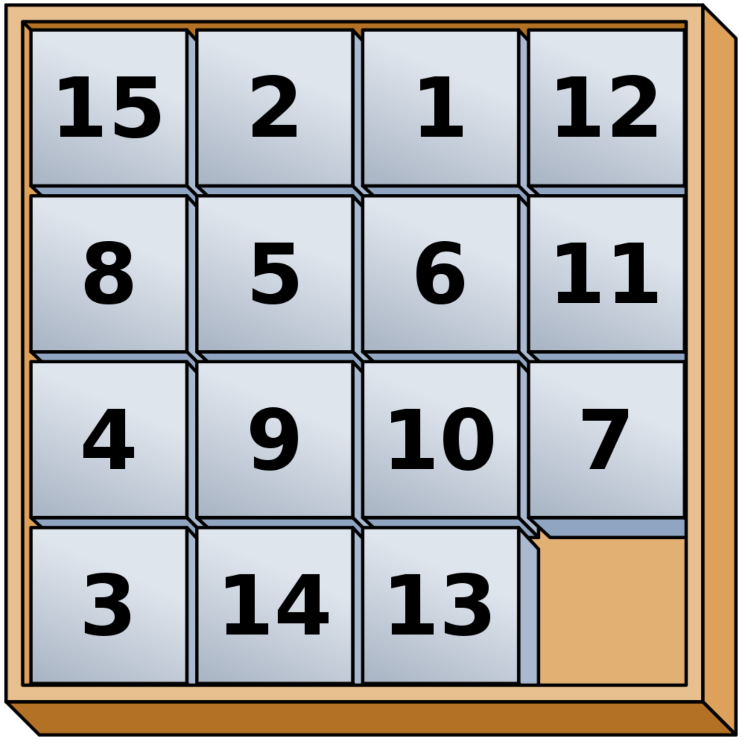
- real life motivation
- environment abstraction – graph (with constant distances)
- goal – plan of movements



INTRODUCTION

MAPF – multi-agent path finding

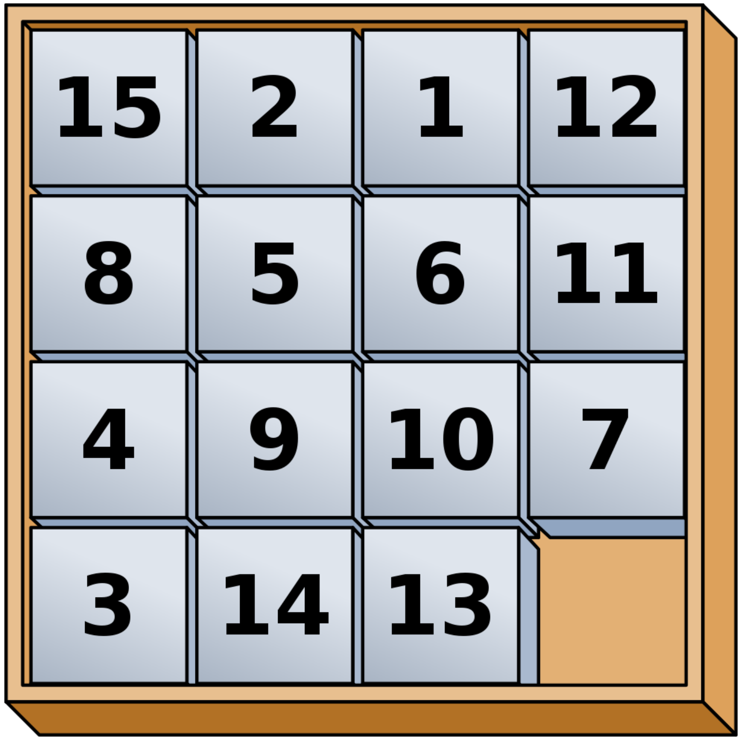
- real life motivation
- environment abstraction – graph (with constant distances)
- goal – plan of movements
- state-space search



INTRODUCTION

MAPF – multi-agent path finding

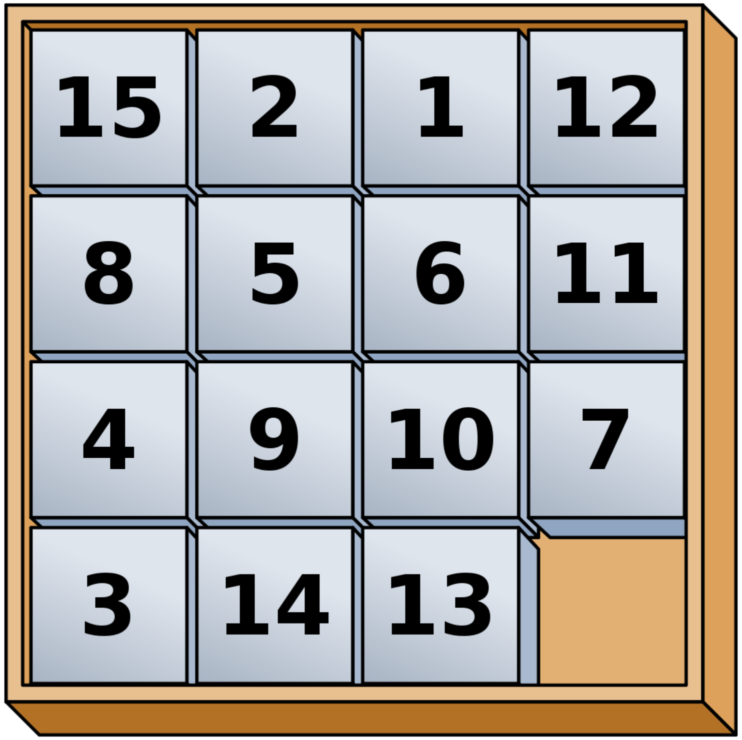
- real life motivation
- environment abstraction – graph (with constant distances)
- goal – plan of movements
- state-space search
- Boolean satisfiability



INTRODUCTION

MAPF – multi-agent path finding

- real life motivation
- environment abstraction – graph (with constant distances)
- goal – plan of movements
- state-space search
- Boolean satisfiability
- optimality:
  - Makespan
  - Sum of Costs



MAPF

PROBLEM DEFINITION

- "finding a collision-free paths for a set of agents"

MAPF

PROBLEM DEFINITION

- "finding a collision-free paths for a set of agents"

- pair (G, A)
  - graph $G = (V, E)$
  - set of agents $A = \{(s_i, g_i)\}$
    - $s, g \in V$

MAPF

PROBLEM DEFINITION

- "finding a collision-free paths for a set of agents"

- pair (G, A)
  - graph $G = (V, E)$
  - set of agents $A = \{(s_i, g_i)\}$
    - $s, g \in V$

- discretized time – time steps

MAPF
PROBLEM DEFINITION

- “finding a collision-free paths for a set of agents”
- pair (G, A)
  - graph $G = (V, E)$
  - set of agents $A = \{(s_i, g_i)\}$
    - $s, g \in V$
- discretized time – time steps
- agent actions – *move* or *wait*
  - *in each time step*

MAPF
PROBLEM DEFINITION

- "finding a collision-free paths for a set of agents"
- pair (G, A)
  - graph $G = (V, E)$
  - set of agents $A = \{(s_i, g_i)\}$
    - $s, g \in V$
- discretized time – time steps
- agent actions – *move* or *wait*
  - *in each time step*
- task – find valid plan for each agent
  - sequence of actions
  - sequence of locations

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

- valid solution of MAPF problem is a plan:

$$\pi = \bigcup_{a_i \in A} \pi_i$$

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

- valid solution of MAPF problem is a plan:

$$\pi = \bigcup_{a_i \in A} \pi_i$$

1. plan for each agent is a valid path

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

- valid solution of MAPF problem is a plan:

$$\pi = \bigcup_{a_i \in A} \pi_i$$

1. plan for each agent is a valid path
2. only one agent can occupy one position at the time
   - plain existence

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

- valid solution of MAPF problem is a plan:

$$\pi = \bigcup_{a_i \in A} \pi_i$$

1. plan for each agent is a valid path
2. only one agent can occupy one position at the time
   - plain existence
3. only one agent can occupy one edge at the time
   - no swap

MAPF

PROBLEM DEFINITION

- π_i plan for agent a_i
- $\pi_i(t)$ location of agent at time t

- valid solution of MAPF problem is a plan:

$$\pi = \bigcup_{a_i \in A} \pi_i$$

1. plan for each agent is a valid path
2. only one agent can occupy one position at the time
   - plain existence
3. only one agent can occupy one edge at the time
   - no swap

Note: train allowed

MAPF

PROBLEM DEFINITION

- Optimal solution – feasible solution with minimal cost

MAPF
PROBLEM DEFINITION

- Optimal solution – feasible solution with minimal cost

Cost functions:

- Makespan $$Mks(\pi) = \max_{i=1\ldots n} |\pi_i|$$
- Sum of Costs $$SoC(\pi) = \sum_{i=1}^{n} |\pi_i|$$

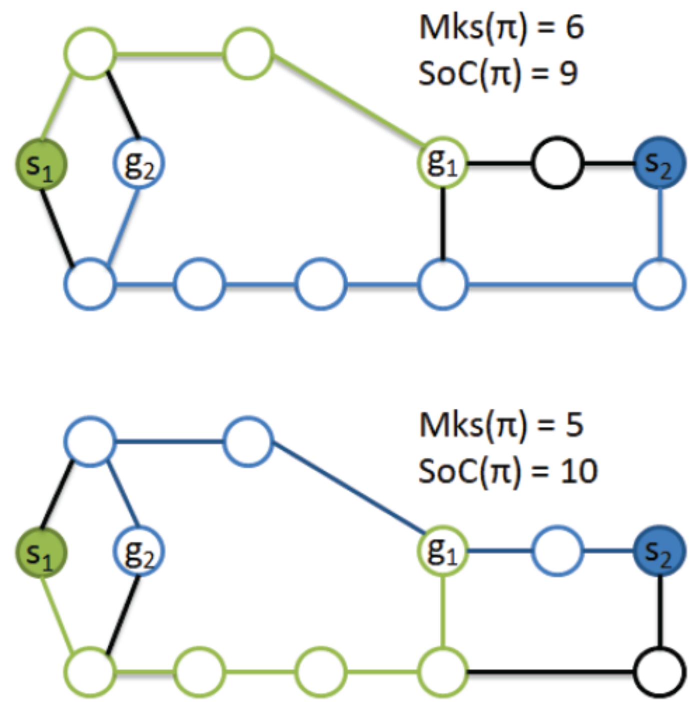
MAPF
PROBLEM DEFINITION

- Optimal solution – feasible solution with minimal cost

Cost functions:

- Makespan $Mks(\pi) = \max_{i=1\ldots n} |\pi_i|$

- Sum of Costs $SoC(\pi) = \sum_{i=1}^{n} |\pi_i|$



MAPF
PROBLEM DEFINITION

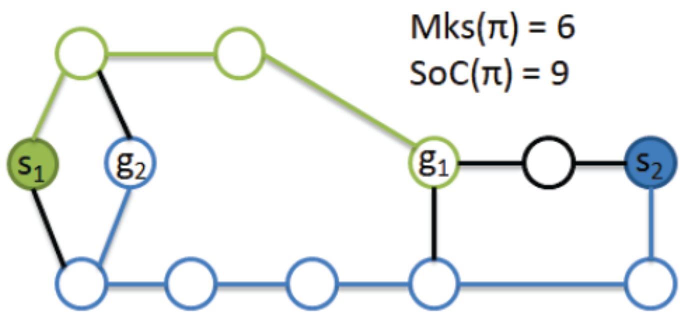
- Optimal solution – feasible solution with minimal cost

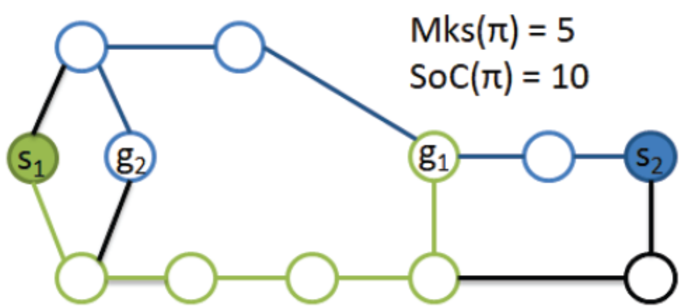
Cost functions:

- Makespan $Mks(\pi) = \max_{i=1\ldots n} |\pi_i|$

- Sum of Costs $SoC(\pi) = \sum_{i=1}^{n} |\pi_i|$

- Feasible solution – polynomial
- Optimal solution – NP-Hard





MAKESPAN OPTIMAL MODEL

- path length is unknown in advance
  - restricted plan length
  - iterative increasing

MAKESPAN OPTIMAL MODEL

- path length is unknown in advance
  - restricted plan length
  - iterative increasing

- MAPF to SAT

$$\forall x \in V, \forall a \in A, \forall t \in \{0, \dots, T\}: \ At(x, a, t)$$

$$\forall (x, y) \in E, \forall a \in A, \forall t \in \{0, \dots, T-1\}:$$

$$Pass(x, y, a, t)$$

MAKESPAN OPTIMAL MODEL

- path length is unknown in advance
  - restricted plan length
  - iterative increasing

- MAPF to SAT

$$\forall x \in V, \forall a \in A, \forall t \in \{0, \ldots, T\}: \ At(x, a, t)$$

$$\forall (x, y) \in E, \forall a \in A, \forall t \in \{0, \ldots, T-1\}:$$
$$Pass(x, y, a, t)$$

Note: $\forall x \in V: \ (x, x) \in E$ – *wait*

MAKESPAN OPTIMAL MODEL

- path length is unknown in advance
  - restricted plan length
  - iterative increasing

- MAPF to SAT

$\forall x \in V, \forall a \in A, \forall t \in \{0, \dots, T\}$: $At(x, a, t)$

$\forall (x, y) \in E, \forall a \in A, \forall t \in \{0, \dots, T-1\}$:

$Pass(x, y, a, t)$

Note: $\forall x \in V$: $(x, x) \in E$ – *wait*

$$\forall a \in A : At(s_a, a, 0) = 1 \quad (1)$$

$$\forall a \in A : At(g_a, a, T) = 1 \quad (2)$$

$$\forall a \in A, \forall t \in \{0, \dots, T\} : \sum_{x \in V} At(x, a, t) \leq 1 \quad (3)$$

$$\forall x \in V, \forall t \in \{0, \dots, T\} : \sum_{a \in A} At(x, a, t) \leq 1 \quad (4)$$

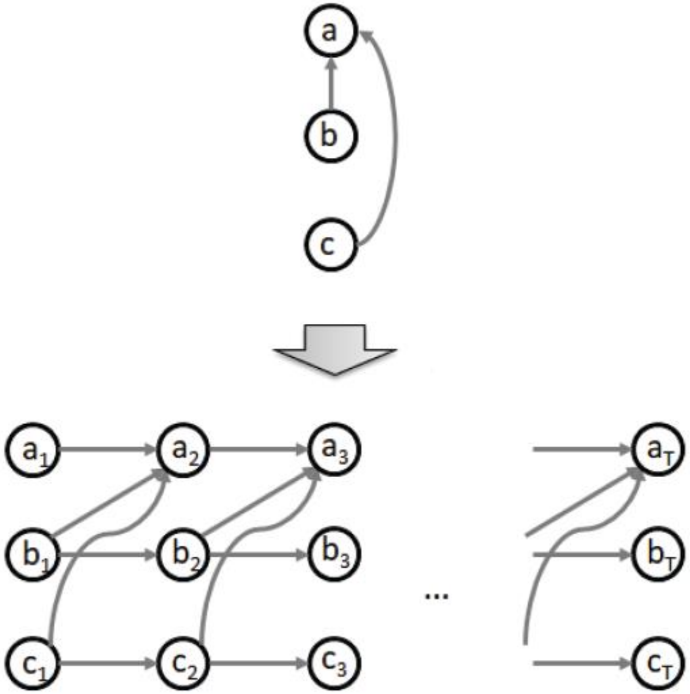
$$\forall x \in V, \forall a \in A, \forall t \in \{0, \dots, T-1\} :$$
$$At(x, a, t) \implies \sum_{(x,y) \in E} Pass(x, y, a, t) = 1 \quad (5)$$

$$\forall (x, y) \in E, \forall a \in A, \forall t \in \{0, \dots, T-1\} :$$
$$Pass(x, y, a, t) \implies At(y, a, t+1) \quad (6)$$

$$\forall (x, y) \in E : x \neq y, \forall t \in \{0, \dots, T-1\} :$$
$$\sum_{a \in A} (Pass(x, y, a, t) + Pass(y, x, a, t)) \leq 1 \quad (7)$$

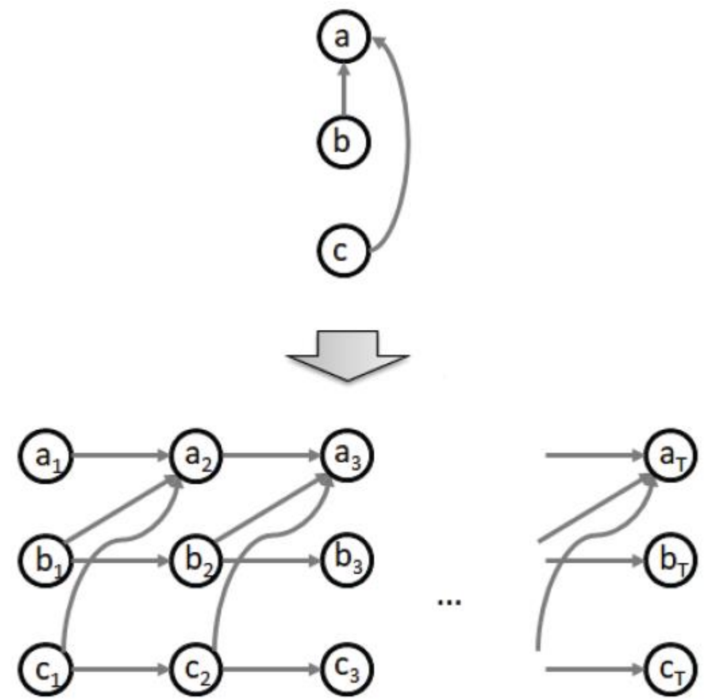
MAKESPAN OPTIMAL MODEL

- SAT representation
  - time-expanded graph



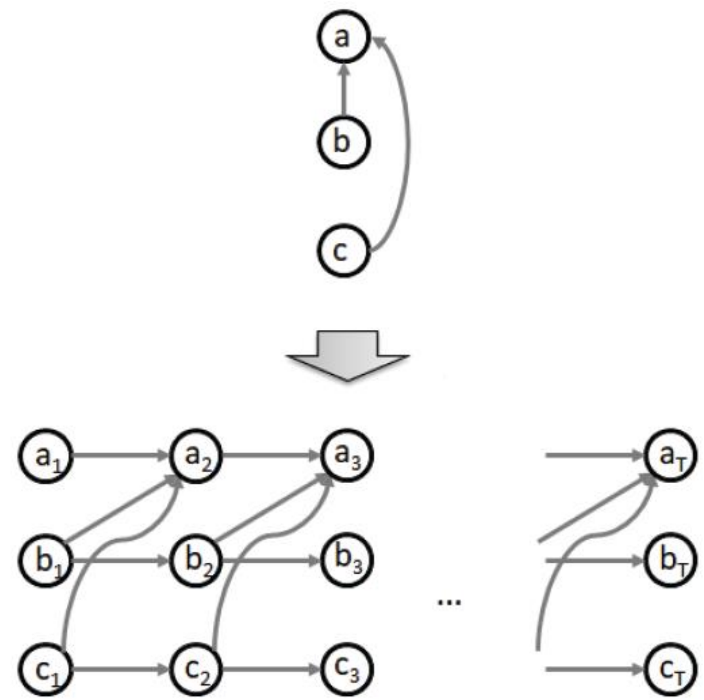
MAKESPAN OPTIMAL MODEL

- SAT representation
  - time-expanded graph
- lower bound
  - $LB(Mks) = \max_{i \in A} SP_i$
  - longest shortest path



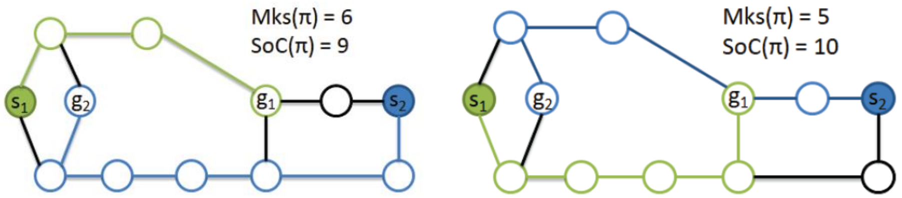
MAKESPAN OPTIMAL MODEL

- SAT representation
  - time-expanded graph
- lower bound
  - $LB(Mks) = \max_{i \in A} SP_i$
  - longest shortest path
- preprocessing for variables
  - some vertices of time expanded graph are for agent unreachable

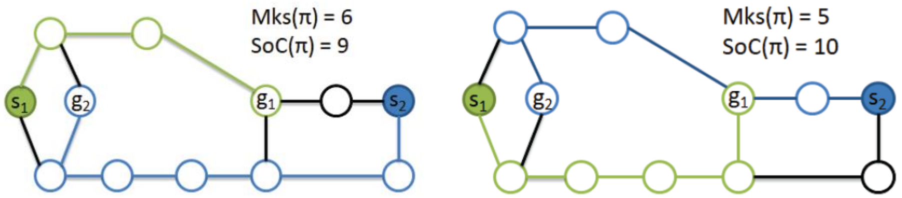


SUM OF COSTS OPTIMAL MODELS

- Makespan approach won't work



SUM OF COSTS OPTIMAL MODELS



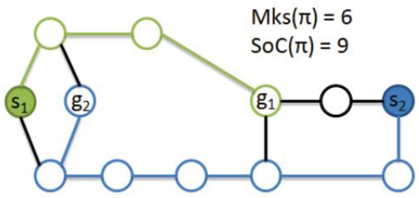
- Makespan approach won't work

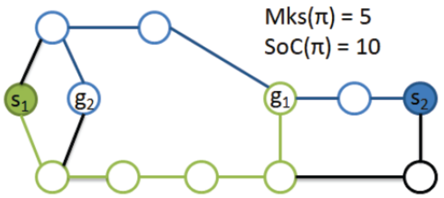
- assuming we can encode upper bound to the model

$$SoC(\pi) \leq UB(SoC) \quad (8)$$

SUM OF COSTS OPTIMAL MODELS

- Makespan approach won't work





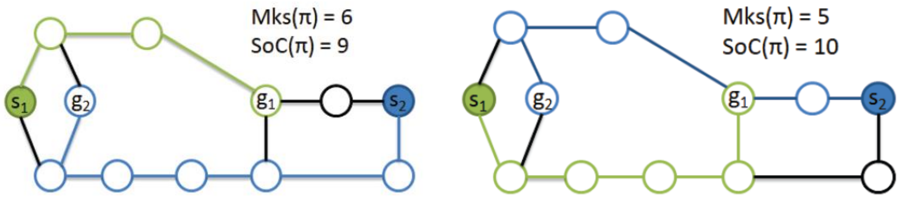
- assuming we can encode upper bound to the model

$$SoC(\pi) \leq UB(SoC) \quad (8)$$

- we should be able to use

$$Minimize\_SoC(LB(SoC), UB(SoC)) \quad (9)$$

SUM OF COSTS OPTIMAL MODELS



- Makespan approach won't work
- assuming we can encode upper bound to the model

$$SoC(\pi) \leq UB(SoC) \quad (8)$$

- we should be able to use

$$Minimize\_SoC(LB(SoC), UB(SoC)) \quad (9)$$

- to obtain plan with lowest SoC in specified interval

MODEL 1
SUM OF COSTS OPTIMAL MODELS

solve_MAPF(T, C)
generates SAT model with:

- constraints 1-7
- Makespan T
- C as UB(SoC)

Algorithm 1 Model 1

function MODEL 1
    $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
    $LB(Mks) = \max_{i \in A} SP_i$
    $LB(SoC) = \sum_{i \in A} SP_i$
    $\delta \leftarrow 0$
    while No Solution **do**
        solve_MAPF($LB(Mks) + \delta, LB(SoC) + \delta$)
        $\delta \leftarrow \delta + 1$
    end while
end function

MODEL 1

SUM OF COSTS OPTIMAL MODELS

solve_MAPF(T, C)
generates SAT model with:

- constraints 1-7
- Makespan T
- C as UB(SoC)

- simultaneously adds
  - layers of time-expanded graph
  - available actions

Algorithm 1 Model 1

function MODEL 1
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\delta \leftarrow 0$
  while No Solution **do**
    solve_MAPF($LB(Mks) + \delta, LB(SoC) + \delta$)
    $\delta \leftarrow \delta + 1$
  end while
end function

THEOREM
SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs* $LB(SoC) + \delta$ *then this solution can be found in a time-expanded graph with* $LB(Mks) + \delta$ *layers.*

THEOREM

SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs $LB(SoC) + \delta$ then this solution can be found in a time-expanded graph with $LB(Mks) + \delta$ layers.*

- solution with Sum of Costs $LB(SoC)$, $LB(SoC) = \sum SP_i$ exists

THEOREM
SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs $LB(SoC) + \delta$ then this solution can be found in a time-expanded graph with $LB(Mks) + \delta$ layers.*

- solution with Sum of Costs $LB(SoC)$, $LB(SoC) = \sum SP_i$ exists
- this plan used δ extra actions in addition to LB

THEOREM

SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs $LB(SoC) + \delta$ then this solution can be found in a time-expanded graph with $LB(Mks) + \delta$ layers.*

- solution with Sum of Costs $LB(SoC)$, $LB(SoC) = \sum SP_i$ exists
- this plan used δ extra actions in addition to LB
- these action can be used by any agent

THEOREM

SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs $LB(SoC) + \delta$ then this solution can be found in a time-expanded graph with $LB(Mks) + \delta$ layers.*

- solution with Sum of Costs $LB(SoC)$, $LB(SoC) = \sum SP_i$ exists
- this plan used δ extra actions in addition to LB
- these action can be used by any agent

- if agent i uses k extra actions, then we need time-expanded graph with at least $SP_i + k$ layers to model its path

THEOREM
SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs* $LB(SoC) + \delta$ *then this solution can be found in a time-expanded graph with* $LB(Mks) + \delta$ *layers.*

- solution with Sum of Costs $LB(SoC),\ LB(SoC) = \sum SP_i$ exists
- this plan used δ extra actions in addition to LB
- these action can be used by any agent
- if agent i uses k extra actions, then we need time-expanded graph with at least $SP_i + k$ layers to model its path
- worst case it is $LB(Mks) +\ \delta$

THEOREM

SUM OF COSTS OPTIMAL MODELS

Theorem 1. *If there exists a solution with the Sum of Costs $LB(SoC) + \delta$ then this solution can be found in a time-expanded graph with $LB(Mks) + \delta$ layers.*

- solution with Sum of Costs $LB(SoC)$, $LB(SoC) = \sum SP_i$ exists
- this plan used δ extra actions in addition to LB
- these action can be used by any agent
- if agent i uses k extra actions, then we need time-expanded graph with at least $SP_i + k$ layers to model its path
- worst case it is $LB(Mks) + \delta$

□

MODEL II
SUM OF COSTS OPTIMAL MODELS

problems with algorithm 1:

- iterates too many makespans
- at the end final makespan is lager than needed

Algorithm 1 Model 1

---

function MODEL 1
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\delta \leftarrow 0$
  while No Solution **do**
    solve_MAPF($LB(Mks) + \delta, LB(SoC) + \delta$)
    $\delta \leftarrow \delta + 1$
  end while
end function

MODEL II

SUM OF COSTS OPTIMAL MODELS

problems with algorithm 1:

- iterates too many makespans
- at the end final makespan is lager than needed

Algorithm 1 Model 1

function MODEL 1
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\delta \leftarrow 0$
  while No Solution **do**
    solve_MAPF($LB(Mks) + \delta, LB(SoC) + \delta$)
    $\delta \leftarrow \delta + 1$
  end while
end function

- find makespan that guarantees existence of optimal solution

  $LB(SoC) = \sum_{i \in A} SP_i$
  $\gamma \leftarrow 0$
  while No Solution **do**
    $SoC \leftarrow$ opt_MAPF($LB(Mks) + \gamma,$
      $LB(SoC), |A| * LB(Mks) + \gamma$)
    $\gamma \leftarrow \gamma + 1$
  end while
  $\delta \leftarrow SoC - LB(SoC)$
  opt_MAPF($LB(Mks) + \delta, LB(SoC), SoC$)
end function

MODEL II

SUM OF COSTS OPTIMAL MODELS

problems with algorithm 1:

- iterates too many makespans
- at the end final makespan is lager than needed

➢ find makespan that guarantees existence of optimal solution

Algorithm 1 Model 1

function MODEL 1
$\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
$LB(Mks) = \max_{i \in A} SP_i$
$LB(SoC) = \sum_{i \in A} SP_i$
$\delta \leftarrow 0$
while No Solution **do**

Algorithm 2 Model 2

function MODEL 2
$\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
$LB(Mks) = \max_{i \in A} SP_i$
$LB(SoC) = \sum_{i \in A} SP_i$
$\gamma \leftarrow 0$
while No Solution **do**
$SoC \leftarrow$ opt_MAPF($LB(Mks) + \gamma$, $LB(SoC), |A| * LB(Mks) + \gamma$)
$\gamma \leftarrow \gamma + 1$
end while
$\delta \leftarrow SoC - LB(SoC)$
opt_MAPF($LB(Mks) + \delta, LB(SoC), SoC$)
end function

MODEL II

SUM OF COSTS OPTIMAL MODELS

1. optimal Makespan is found
   - with no restriction on Sum of Costs

Algorithm 2 Model 2

function MODEL 2
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\gamma \leftarrow 0$
  while No Solution **do**
    $SoC \leftarrow$ opt_MAPF($LB(Mks) + \gamma$,
        $LB(SoC), |A| * LB(Mks) + \gamma$)
    $\gamma \leftarrow \gamma + 1$
  end while
  $\delta \leftarrow SoC - LB(SoC)$
  opt_MAPF($LB(Mks) + \delta, LB(SoC), SoC$)
end function

MODEL II

SUM OF COSTS OPTIMAL MODELS

1. optimal Makespan is found
   - with no restriction on Sum of Costs
2. computes δ by Theorem 1

Algorithm 2 Model 2

function MODEL 2
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\gamma \leftarrow 0$
  while No Solution **do**
    $SoC \leftarrow$ opt_MAPF($LB(Mks) + \gamma$,
      $LB(SoC), |A| * LB(Mks) + \gamma$)
    $\gamma \leftarrow \gamma + 1$
  end while
  $\delta \leftarrow SoC - LB(SoC)$
  opt_MAPF($LB(Mks) + \delta, LB(SoC), SoC$)
end function

MODEL II
SUM OF COSTS OPTIMAL MODELS

1. optimal Makespan is found
   - with no restriction on Sum of Costs
2. computes δ by Theorem 1
3. finds optimal solution

Algorithm 2 Model 2

```
function MODEL 2
    ∀a_i ∈ A : SP_i = shortest_path(s_i, g_i)
    LB(Mks) = max_{i∈A} SP_i
    LB(SoC) = Σ_{i∈A} SP_i
    γ ← 0
    while No Solution do
        SoC ← opt_MAPF(LB(Mks) + γ,
                LB(SoC), |A| ∗ LB(Mks) + γ)
        γ ← γ + 1
    end while
    δ ← SoC − LB(SoC)
    opt_MAPF(LB(Mks) + δ, LB(SoC), SoC)
end function
```

MODEL II

SUM OF COSTS OPTIMAL MODELS

1. optimal Makespan is found
   - with no restriction on Sum of Costs
2. computes δ by Theorem 1
3. finds optimal solution

opt_MAPF(T, L, U)
generates SAT model with:

- constraints 1-7
- Makespan T
- L as LB(SoC)
- U as UB(SoC)

Algorithm 2 Model 2

function MODEL 2
  $\forall a_i \in A : SP_i = shortest\_path(s_i, g_i)$
  $LB(Mks) = \max_{i \in A} SP_i$
  $LB(SoC) = \sum_{i \in A} SP_i$
  $\gamma \leftarrow 0$
  while No Solution **do**
    $SoC \leftarrow$ opt_MAPF($LB(Mks) + \gamma$,
        $LB(SoC), |A| * LB(Mks) + \gamma$)
    $\gamma \leftarrow \gamma + 1$
  end while
  $\delta \leftarrow SoC - LB(SoC)$
  opt_MAPF($LB(Mks) + \delta, LB(SoC), SoC$)
end function

MODEL II

SUM OF COSTS OPTIMAL MODELS

1. optimal Makespan is found
   - with no restriction on Sum of Costs
2. computes δ by Theorem 1
3. finds optimal solution

opt_MAPF(T, L, U)
generates SAT model with:

- constraints 1-7
- Makespan T
- L as LB(SoC)
- U as UB(SoC)

Algorithm 2 Model 2

```
function MODEL 2
    ∀a_i ∈ A : SP_i = shortest_path(s_i, g_i)
    LB(Mks) = max_{i∈A} SP_i
    LB(SoC) = Σ_{i∈A} SP_i
    γ ← 0
    while No Solution do
        SoC ← opt_MAPF(LB(Mks) + γ,
                 LB(SoC), |A| ∗ LB(Mks) + γ)
        γ ← γ + 1
    end while
    δ ← SoC − LB(SoC)
    opt_MAPF(LB(Mks) + δ, LB(SoC), SoC)
end function
```

- optimal solution found using (9) in interval:

$$\langle LB(SoC), |A| * LB(Mks) + \gamma \rangle$$

$$Minimize\_SoC(LB(SoC), UB(SoC)) \quad (9)$$

MODEL II

SUM OF COSTS OPTI

- with best Sum of Cost from all optimal Makespans
- any feasible Makespan is sufficient at this stage, but nonoptimal is costly

Algor

1. optimal Makespan is found
   - with no restriction on Sum of Costs
2. computes δ by Theorem 1
3. finds optimal solution

opt_MAPF(T, L, U)
generates SAT model with:

- constraints 1-7
- Makespan T
- L as LB(SoC)
- U as UB(SoC)

w

$SoC \leftarrow$ opt_MAPF$(LB(Mks) + \gamma,$
$LB(SoC), |A| * LB(Mks) + \gamma)$
$\gamma \leftarrow \gamma + 1$
end while
$\delta \leftarrow SoC - LB(SoC)$
opt_MAPF$(LB(Mks) + \delta, LB(SoC), SoC)$
end function

- optimal solution found using (9) in interval:

$$\langle LB(SoC), |A| * LB(Mks) + \gamma \rangle$$

$$Minimize\_SoC(LB(SoC), UB(SoC)) \qquad (9)$$

REDUCTION OF USED VARIABLES

SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

REDUCTION OF USED VARIABLES

SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers

REDUCTION OF USED VARIABLES
SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"

REDUCTION OF USED VARIABLES
SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"
- agents with short shortest path have no chance of reaching all levels

REDUCTION OF USED VARIABLES
SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"
- agents with short shortest path have no chance of reaching all levels

➢ only when solving Sum of Costs!

REDUCTION OF USED VARIABLES

SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"
- agents with short shortest path have no chance of reaching all levels

- only when solving Sum of Costs!
- for each agent is created separate time-expanded graph with $SP_i + \delta$ layers

REDUCTION OF USED VARIABLES
SUM OF COSTS OPTIMAL MODELS

time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"
- agents with short shortest path have no chance of reaching all levels

- only when solving Sum of Costs!
- for each agent is created separate time-expanded graph with $SP_i + \delta$ layers
- such graphs are interconnected by constraints 1-7

REDUCTION OF USED VARIABLES
SUM OF COSTS OPTIMAL MODELS

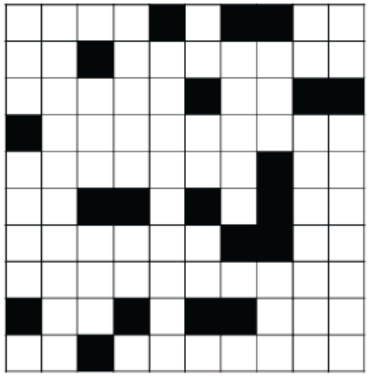
time-expanded graph represented by SAT model can be seen as if each agent had his own version of this graph and those were connected by constraints

- for Makespan all agents needs same numbers of layers
- but for Sum of Costs agents "share available moves"
- agents with short shortest path have no chance of reaching all levels

- only when solving Sum of Costs!
- for each agent is created separate time-expanded graph with $SP_i + \delta$ layers
- such graphs are interconnected by constraints 1-7
- in this way, agents with short paths would disappear from abstract graphs of other players once they reach goal
  - forbid their goal after timestep $SP_i + \delta$

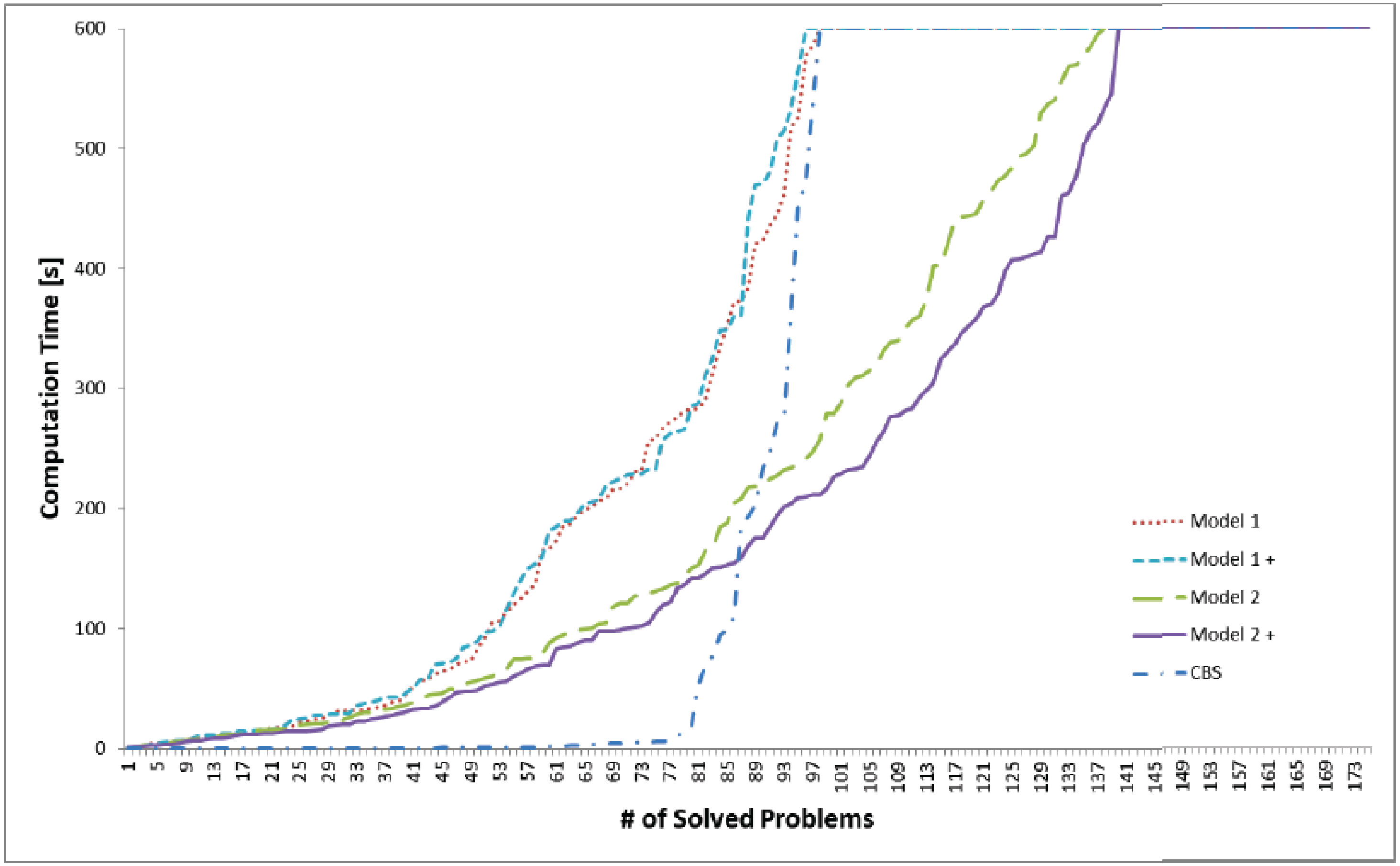
EXPERIMENTS

- 2d grids 8x8 up to 16x16
- 20% of the cells are impassable
- 1-2x grid width agents
- randomly generated unique start and unique end positions



- each setting 5x
- altogether 175 unique instances

EXPERIMENTS



EXPERIMENTS

	M. 1	M. 2	M. 1+	M. 2+	CBS
# of solved	97	137	95	**139**	97
# of fastest	0	4	3	46	**88**
# of fastest (without CBS)	9	8	6	**118**	–
IPC score	16.11	44.56	16.76	57.07	**92.50**
IPC score (without CBS)	57.19	110.54	53.93	**134.29**	–

KEYWORDS

- MAPF
  - SAT representation
    - Time-expanded graph
      - Sum of Costs
        - Makespan
          - Model 1
            - Theorem 1
              - Model 2
                - Reduction of Used Variables