Sufficient Conditions for Node Expansion in Bidirectional Heuristic Search

Jürgen Eckerle[1], Jingwei Chen[2], Nathan R. Sturtevant[2], Sandra Zilles[3] and Robert C. Holte[4]

[1]Berner Fachhochschule
[2]University of Denver
[3]University of Regina
[4]University of Alberta

Moving AI Lab
NSERC CRSNG
NSF Grant 1551406
UNIVERSITY of DENVER
DANIEL FELIX RITCHIE SCHOOL OF ENGINEERING & COMPUTER SCIENCE

Main Result

Main Result

- Unidirectional algorithms:
  - n must be expanded if $f(n) < C^*$
- Bidirectional algorithms:
  - Which set of states must be expanded?
    - No single state must be expanded
    - For a pair of states, if certain condition holds (sufficient condition), then at least one of them must be expanded

Agenda

- Assumptions
- Background
  - Unidirectional Search
  - Bidirectional Search
- Main Result
  - No single state must be expanded
  - Sufficient condition
- Applications and Conclusion

Agenda

- Assumptions
- Background
  - Unidirectional Search
  - Bidirectional Search
- Main Result
  - No single state must be expanded
  - Sufficient condition
- Applications and Conclusion

Assumptions

- Heuristic Search Problems
  - State Space: (start, goal, cost function, successor function, predecessor function)
  - Optimal solution
    - Cost: C*
  - Heuristic function
    - consistent

Assumptions

- Heuristic Search Algorithms
  - Admissible
  - Deterministic, Expansion-based, Black Box (DXBB)

Agenda

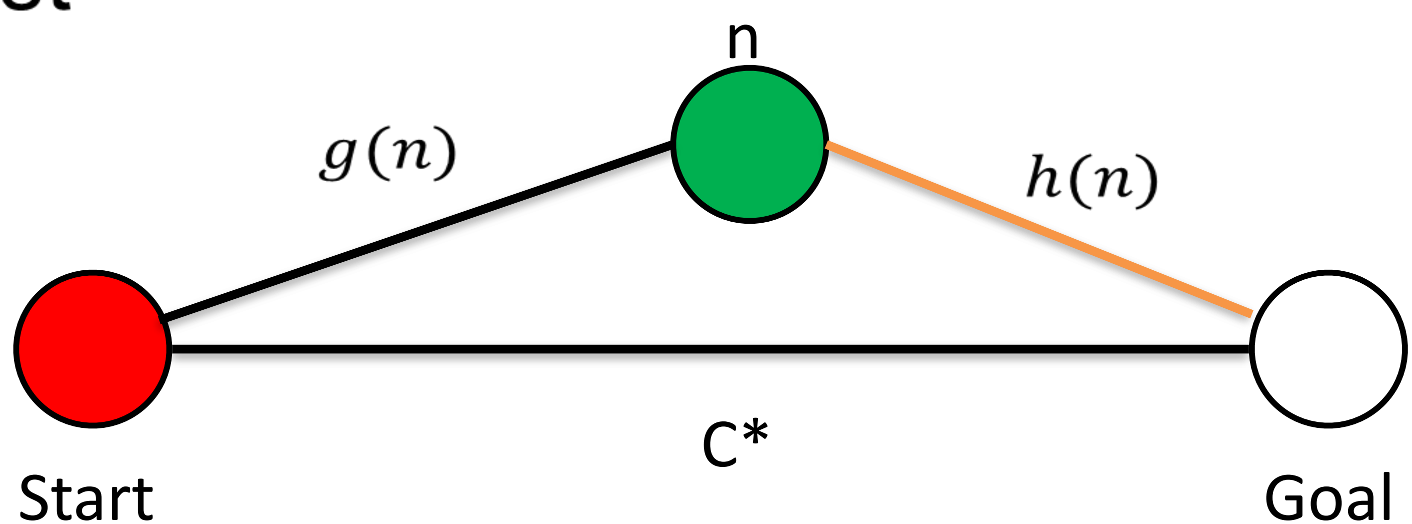
- Assumptions
- Background
  - Unidirectional Search
  - Bidirectional Search
- Main Result
  - No single state must be expanded
  - Sufficient condition
- Applications and Conclusion

Unidirectional Search

- A*
    - Invented in 1968 [Hart, Nilsson and Raphael]
    - Proved to be optimal unidirectional algorithm in 1985 [Dechter & Pearl]

A*

- g-cost: actual cost so far
- h-cost: estimated cost to go (heuristic value)
- $f(u) = g(u) + h(u)$: estimated total path cost



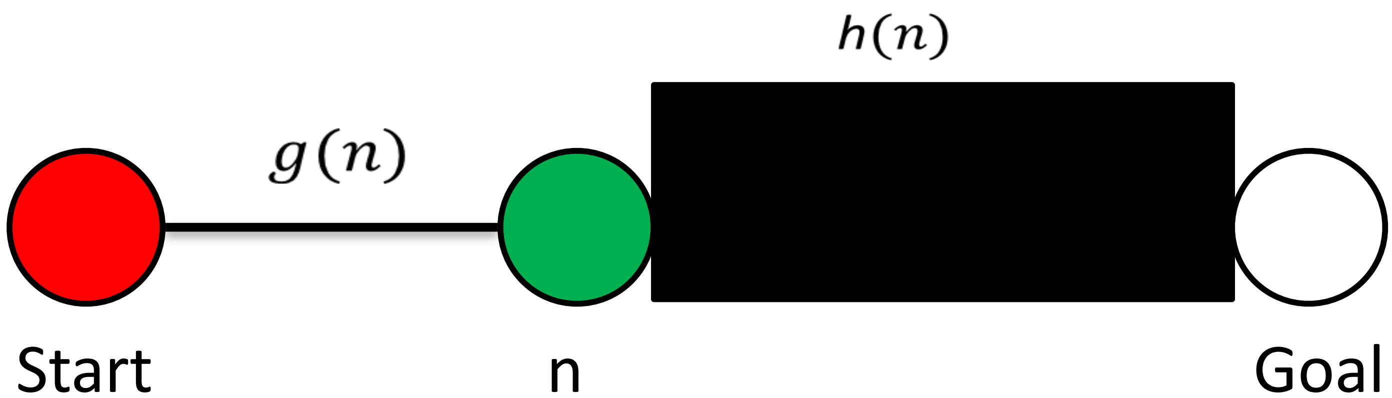
Optimality of A*

- What does optimality mean?
  - For any given consistent heuristic, A* does minimum node expansions (up to tie-breaking)

Optimality of A*

- $f(n) = g(n) + h(n) \; < C^*$

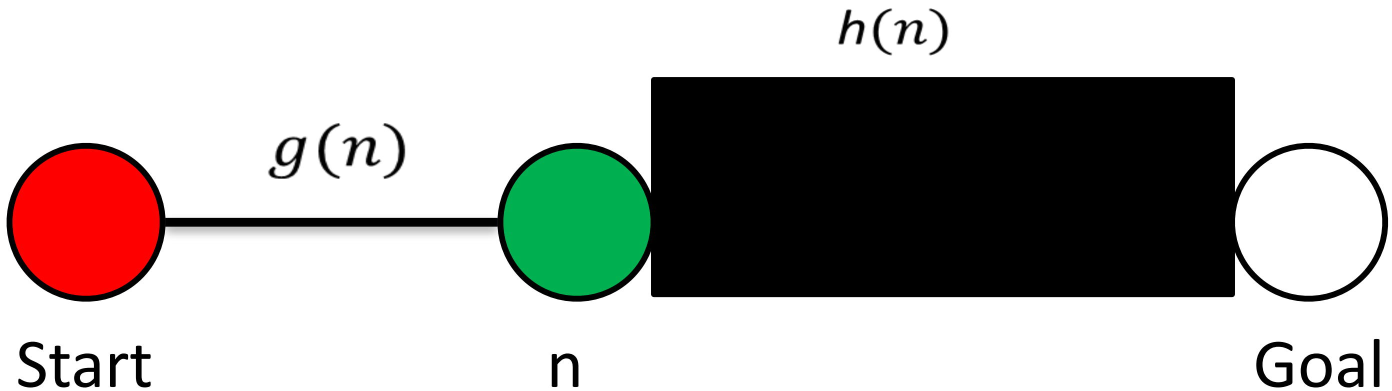
Instance 1:



Optimality of A*

- $f(n) = g(n) + h(n) \ < C^*$

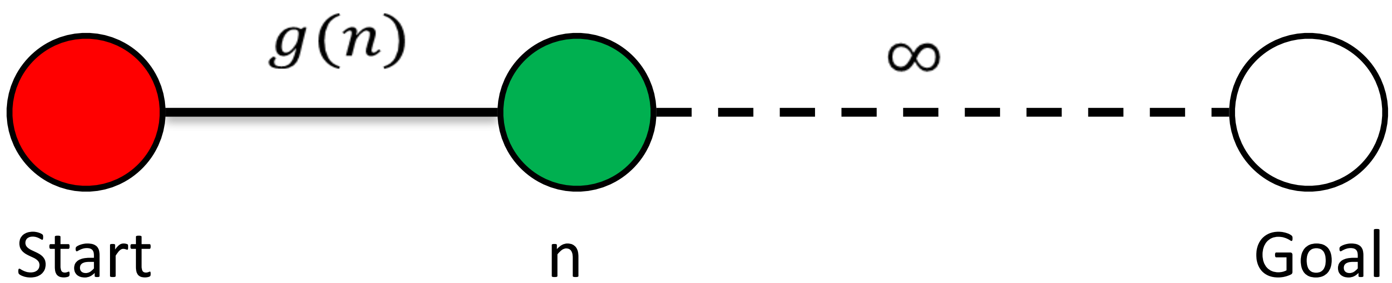
Instance 1:



Optimality of A*

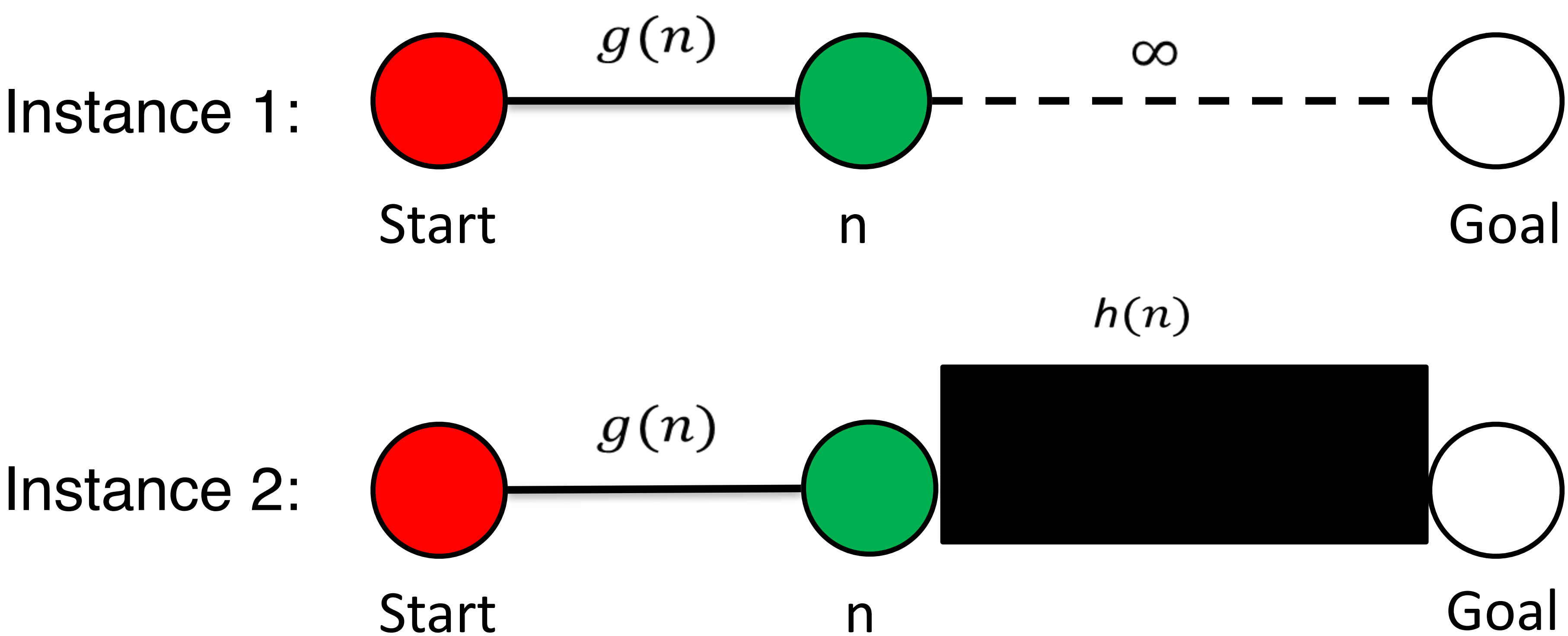
- $f(n) = g(n) + h(n) \; < C^*$

Instance 1:



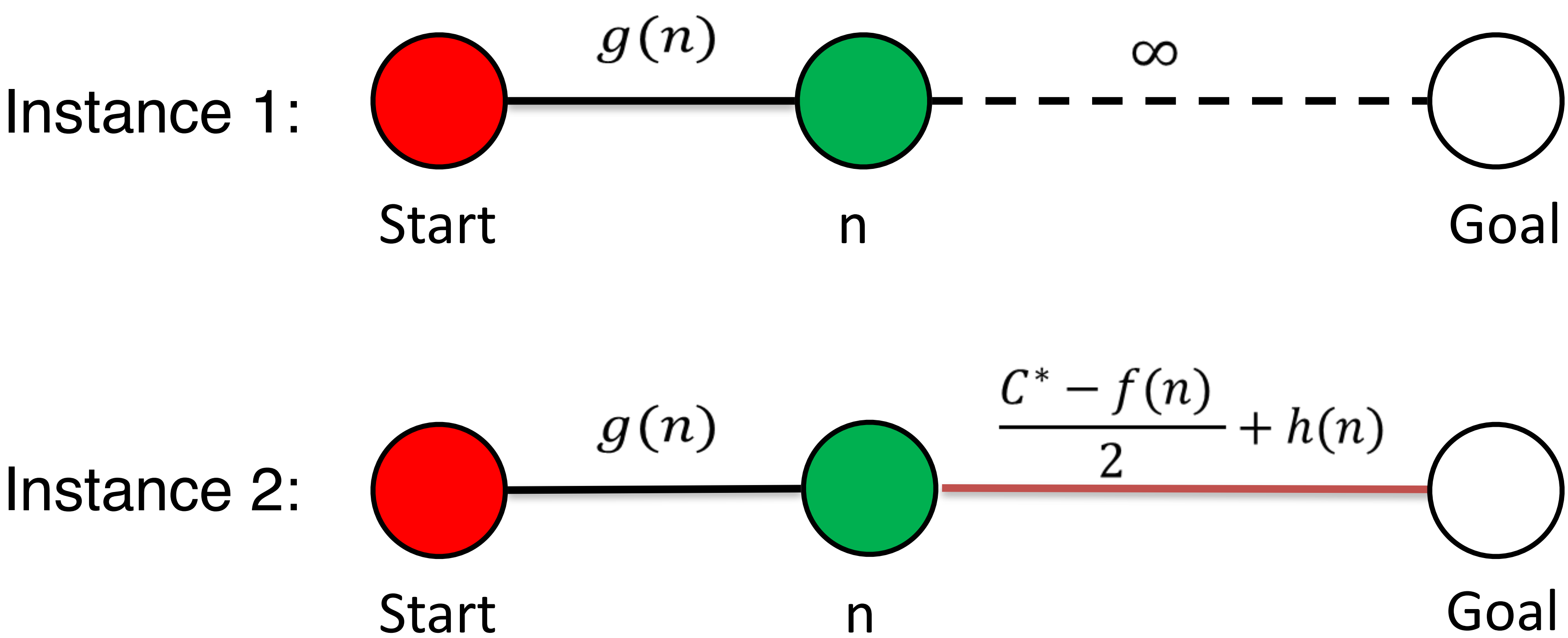
Optimality of A*

- $f(n) = g(n) + h(n) \ < C^*$



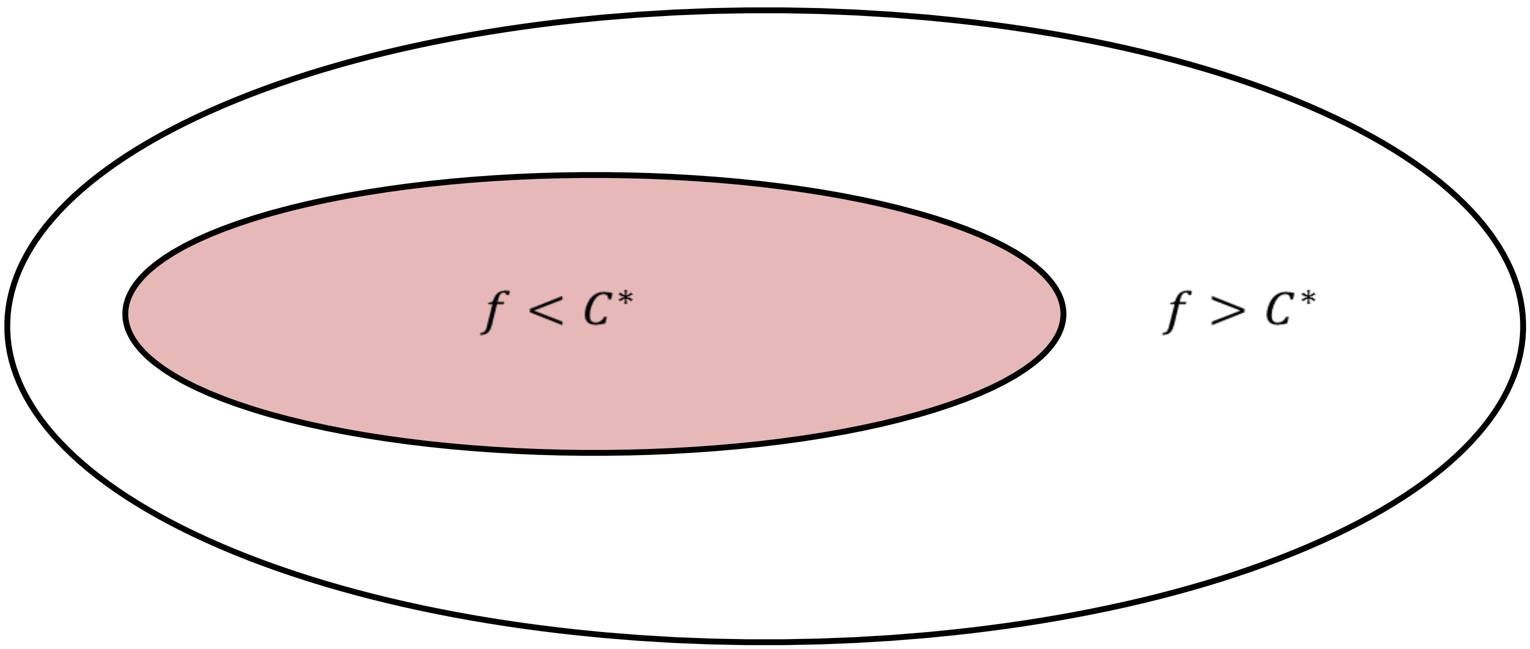
Optimality of A*

- $f(n) = g(n) + h(n) \ < C^*$



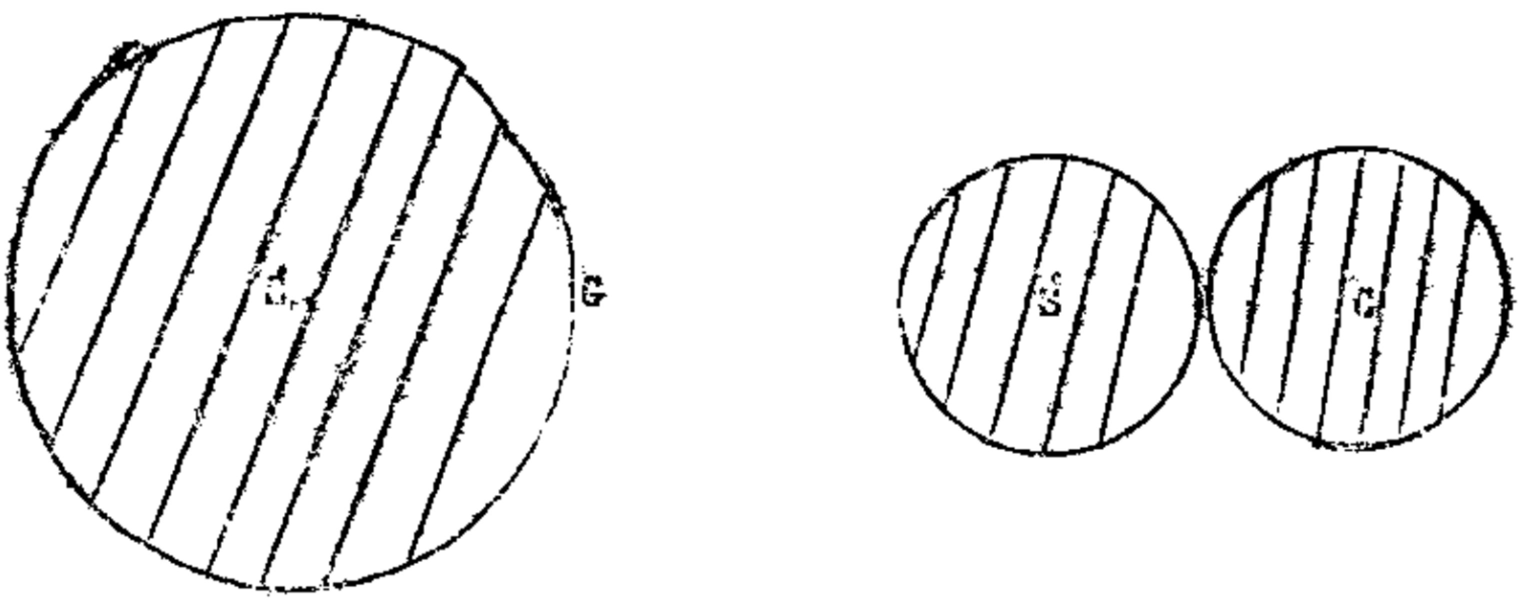
Optimality of A*

- Any algorithm must expand all $f < C^*$
- A* does exactly $f < C^*$
- A* does minimum amount of work.



Bidirectional Search

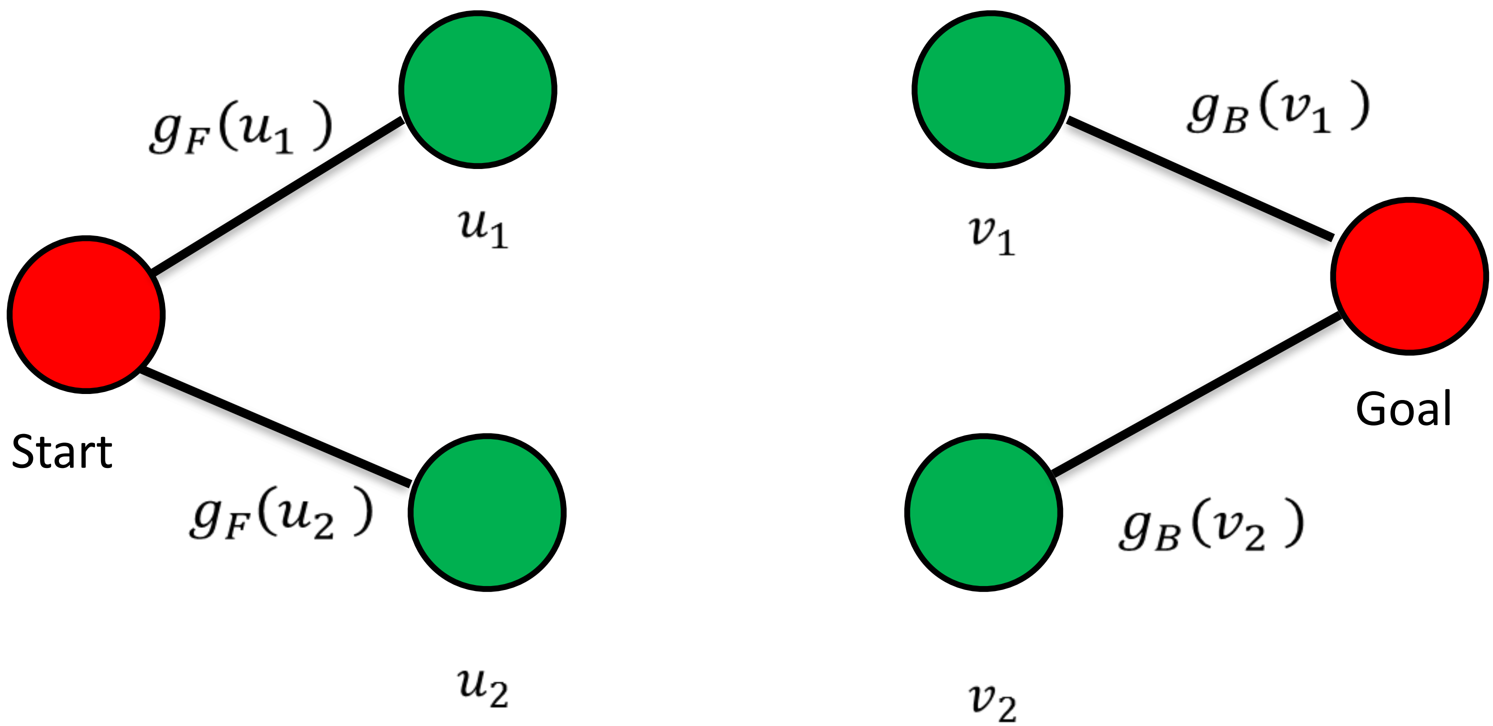
- Bidirectional heuristic search algorithms
  - Have potential to do less work
  - [Nicholson 1966; Doran 1966]



[Figure from Doran 1966]

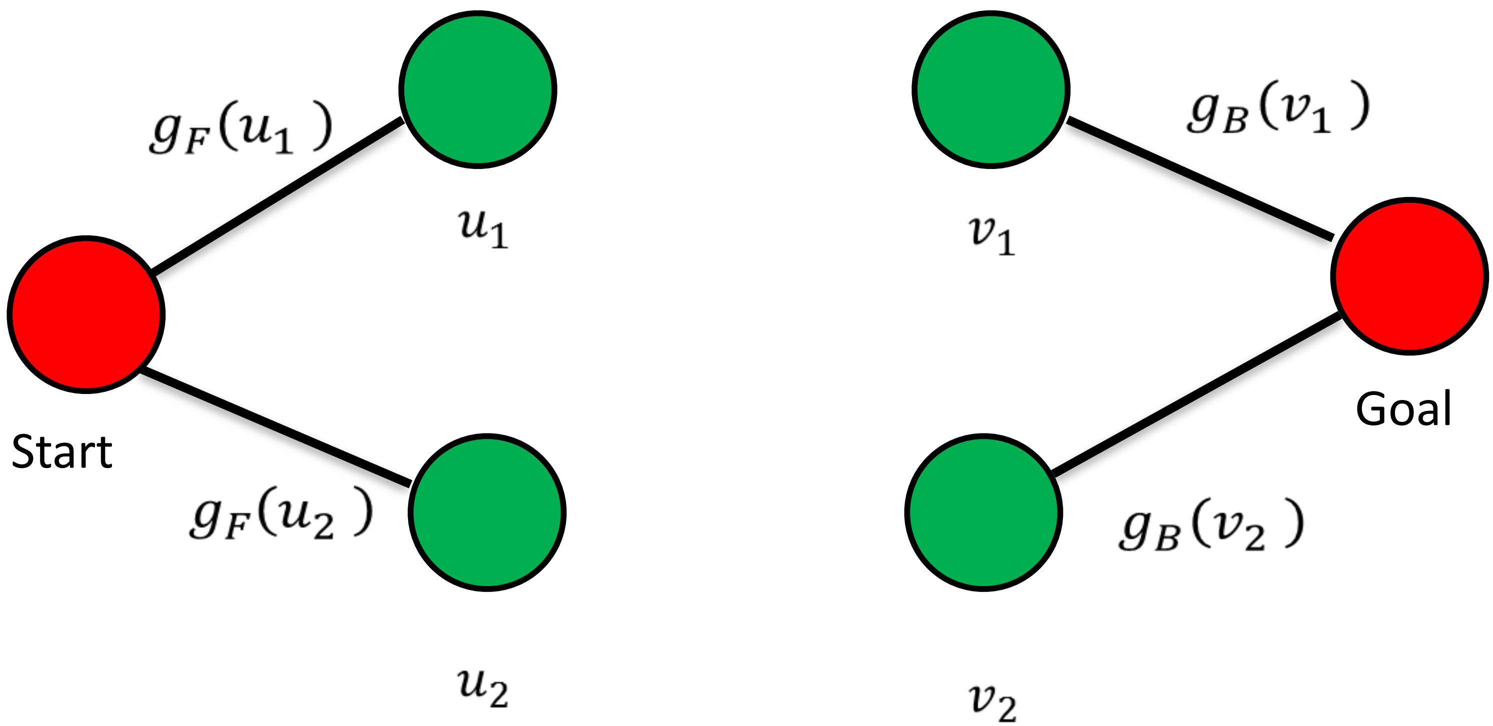
Bidirectional Search

- Front-to-front vs. front-to-end



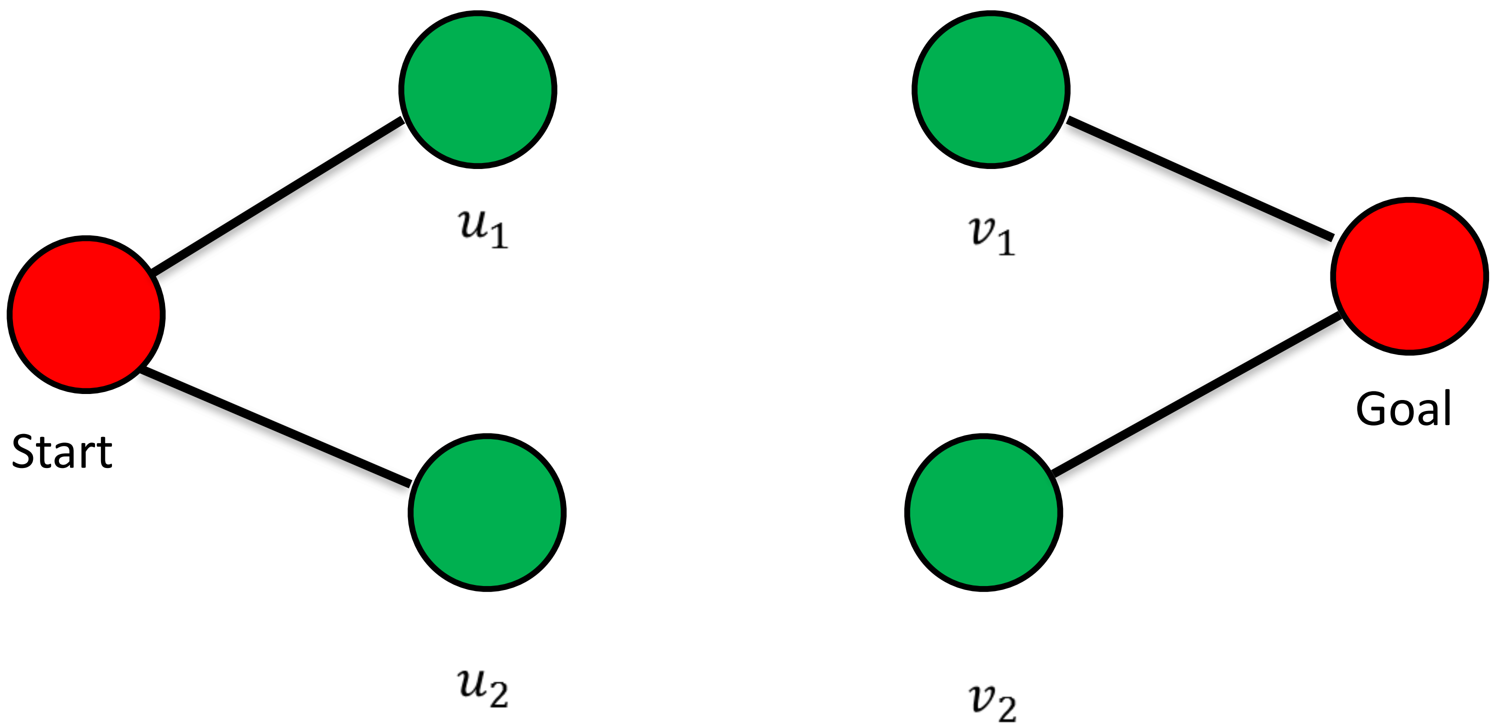
Bidirectional Search

- Front-to-front vs. front-to-end



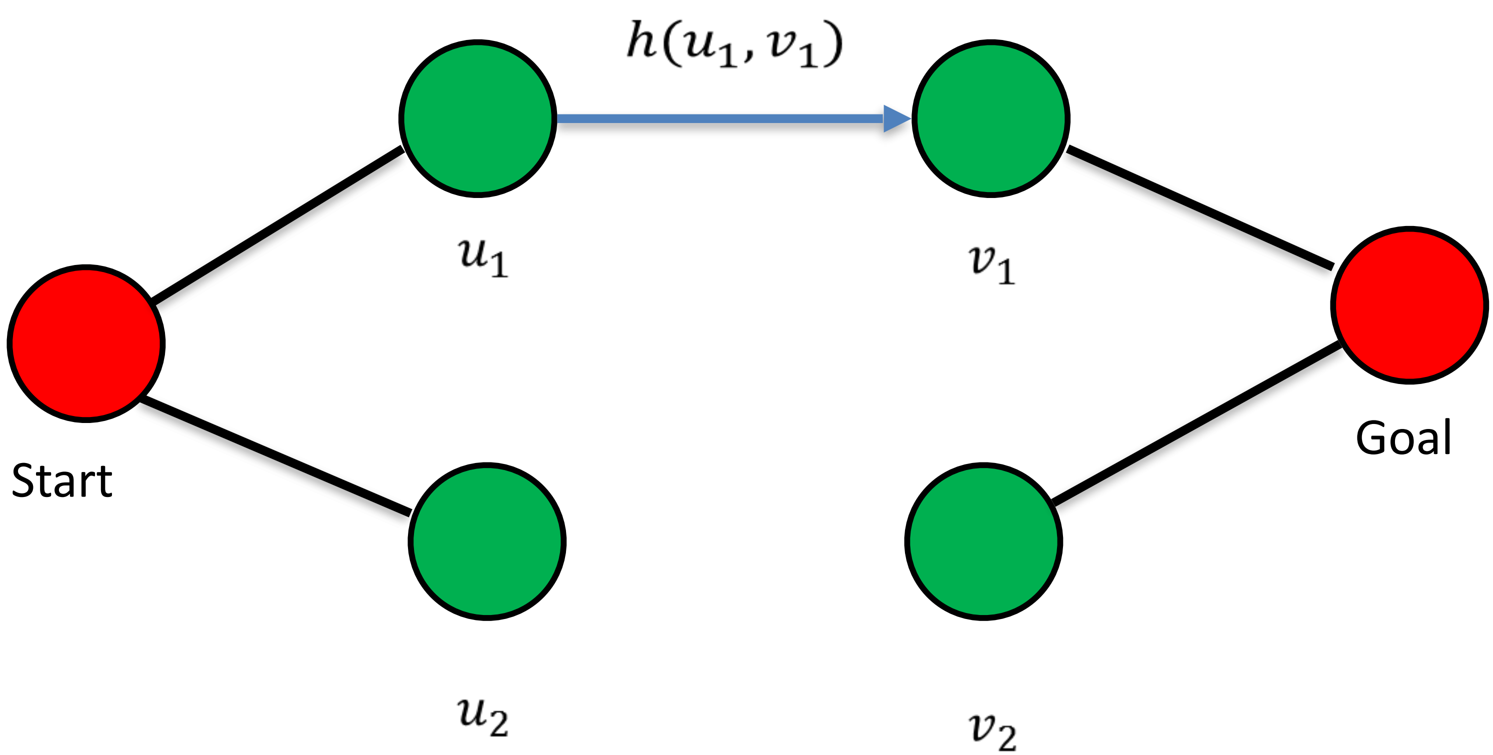
Front-to-Front

- One front-to-front heuristic



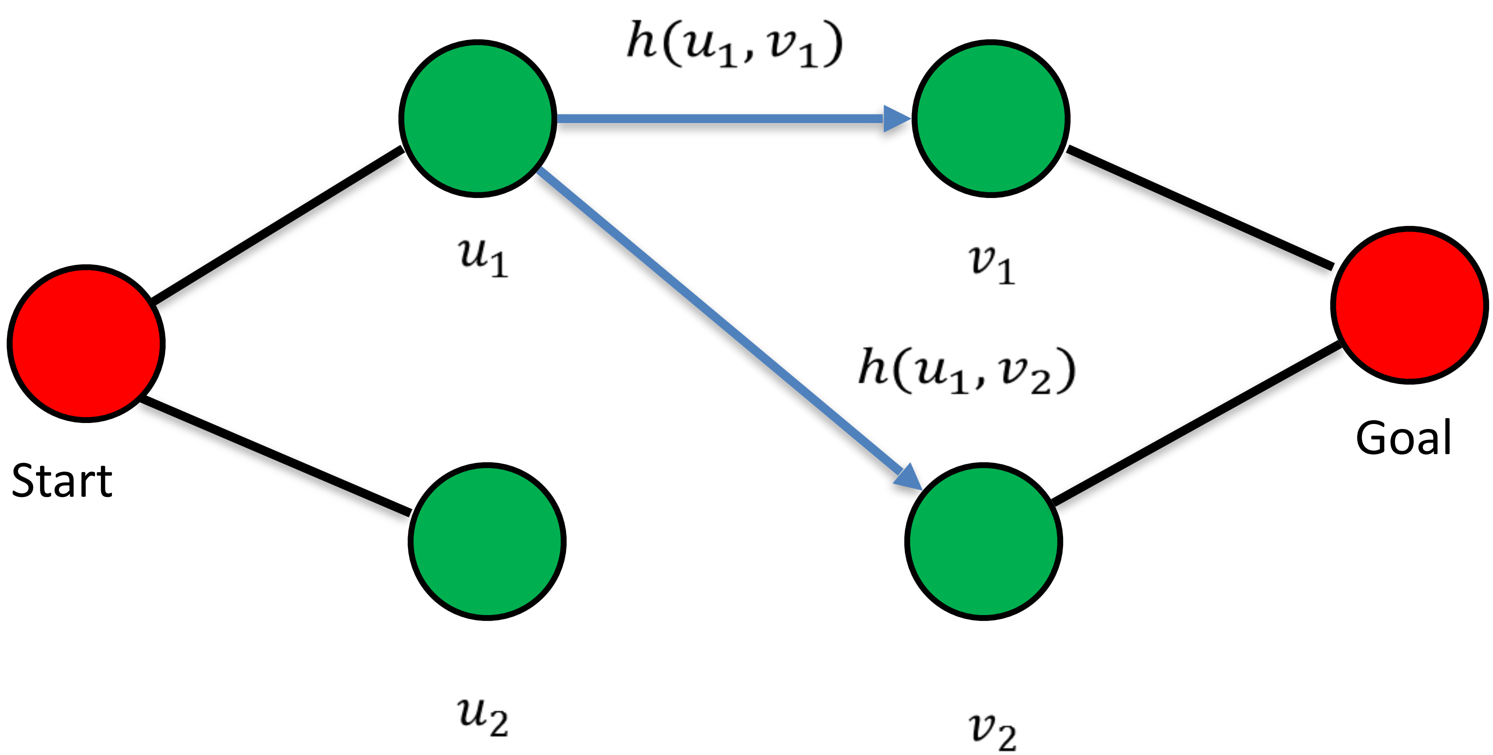
Front-to-Front

- One front-to-front heuristic



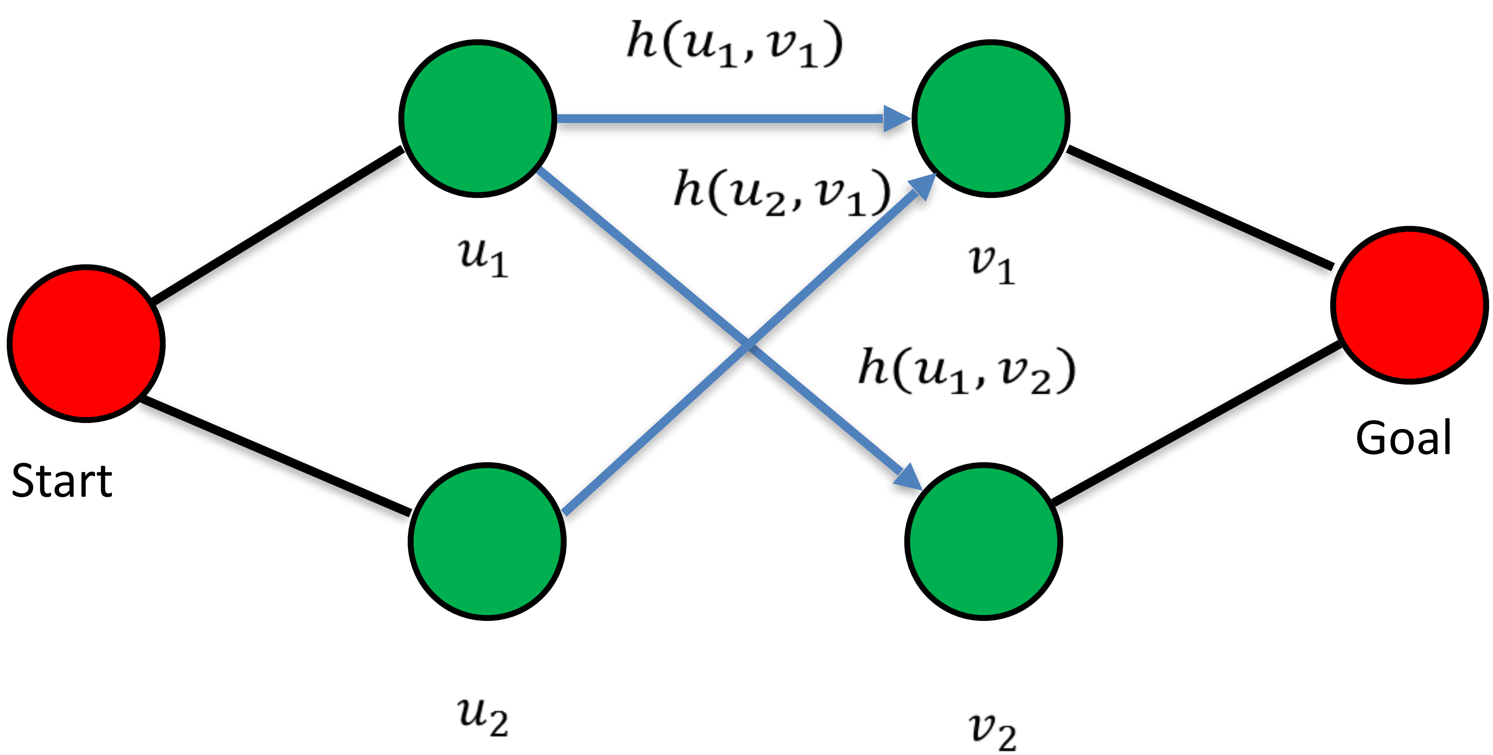
Front-to-Front

- One front-to-front heuristic



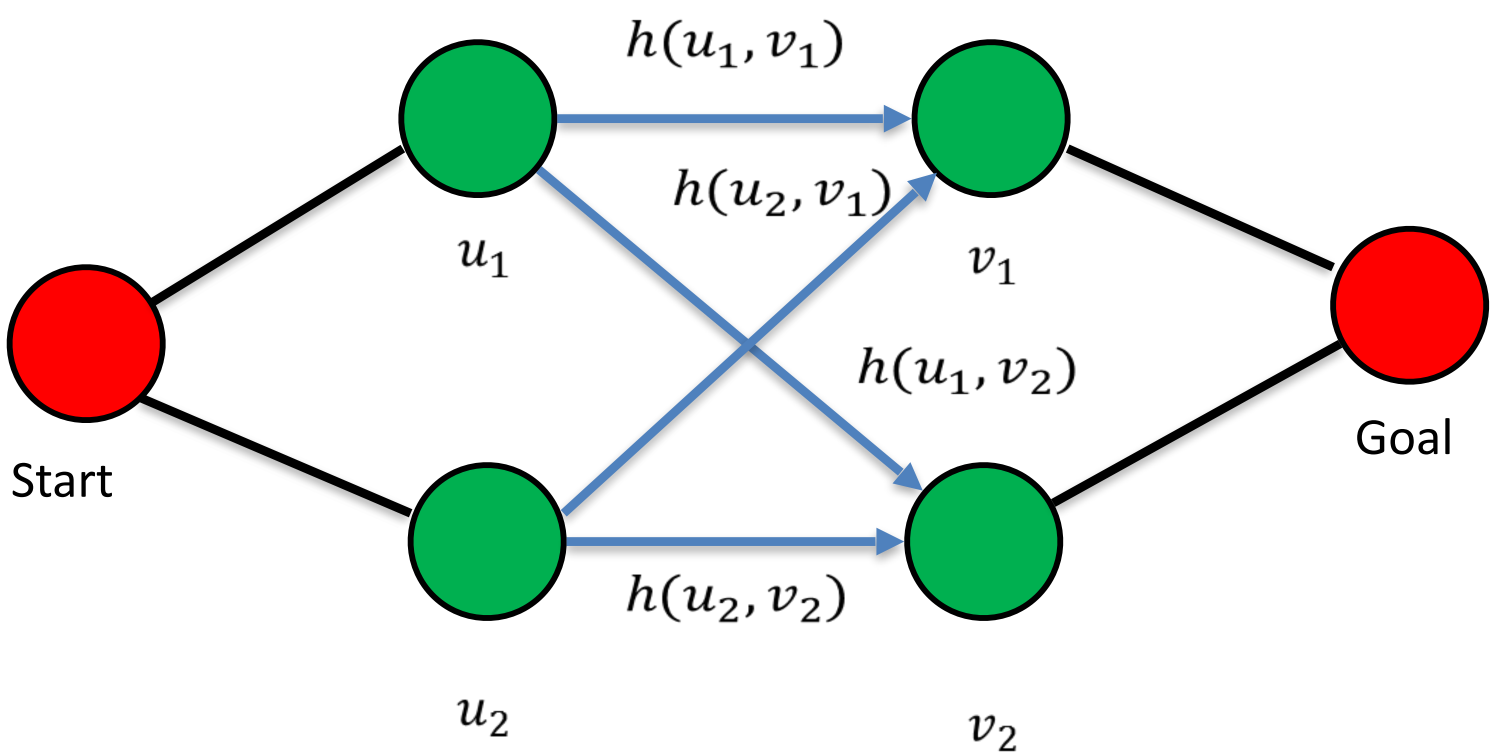
Front-to-Front

- One front-to-front heuristic



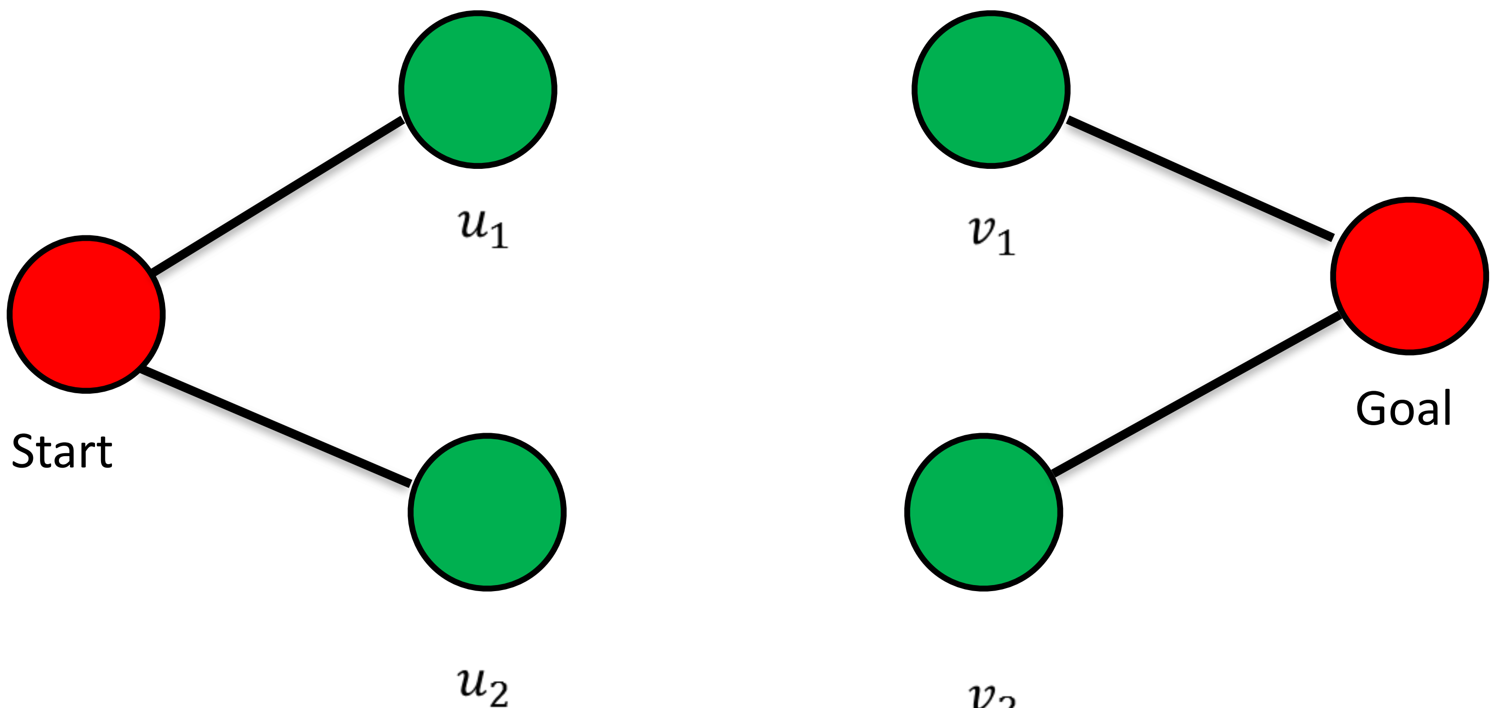
Front-to-Front

- One front-to-front heuristic



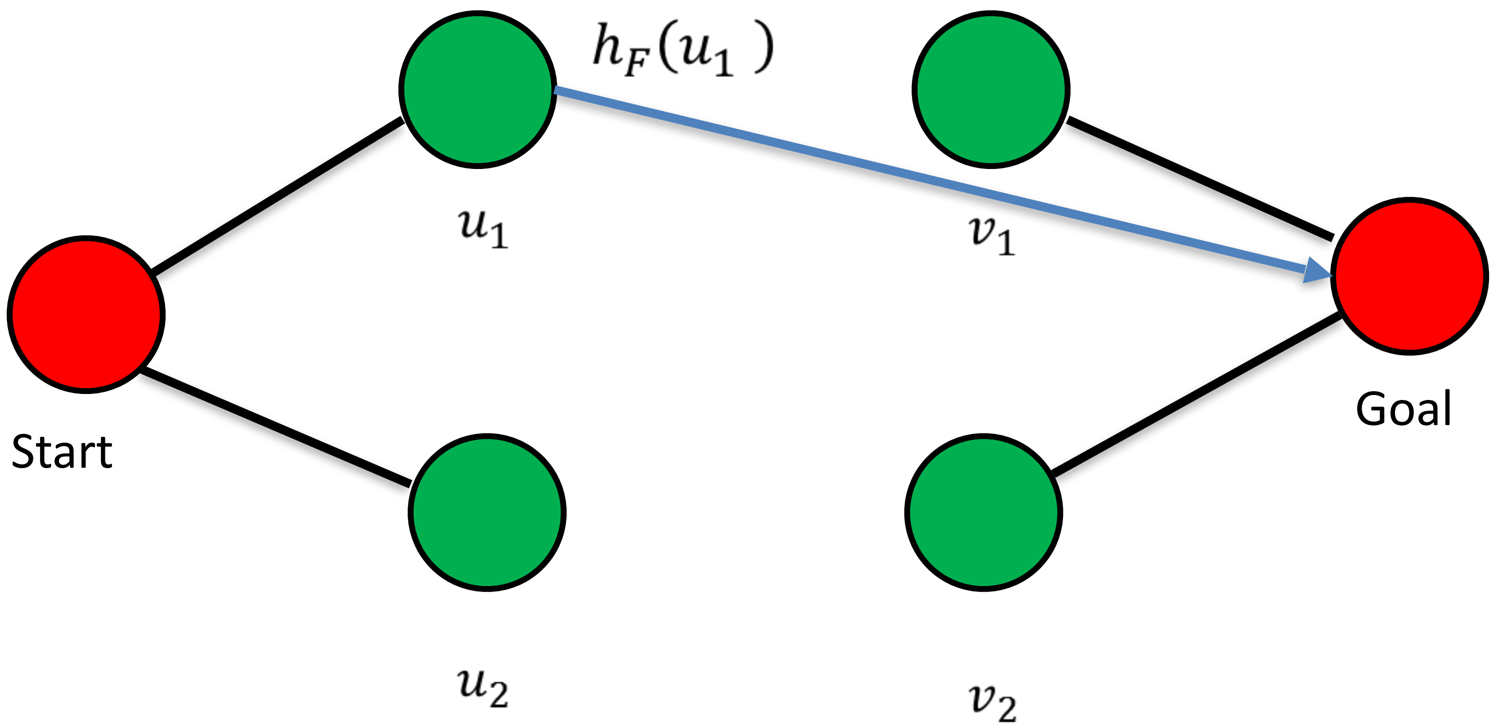
Front-to-End

- Two front-to-end heuristics



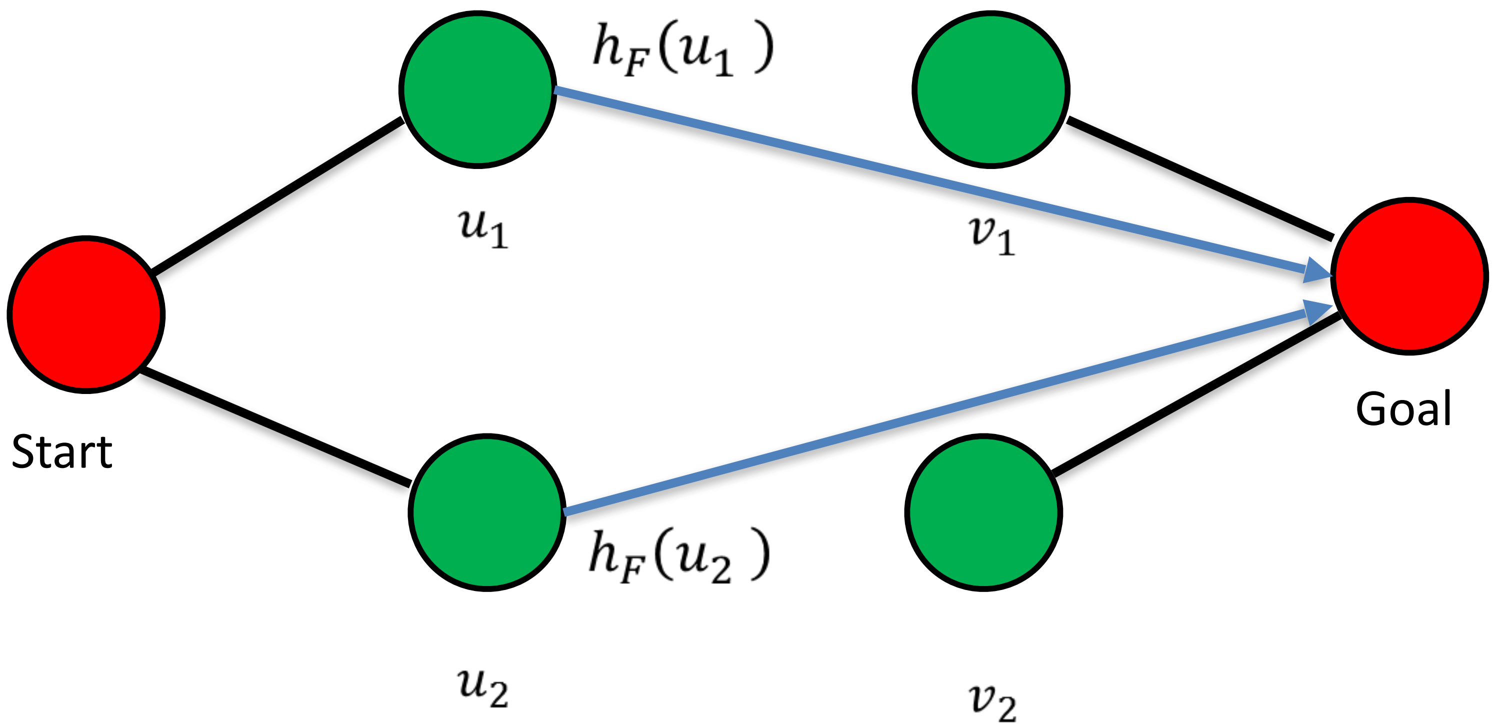
Front-to-End

- Two front-to-end heuristics



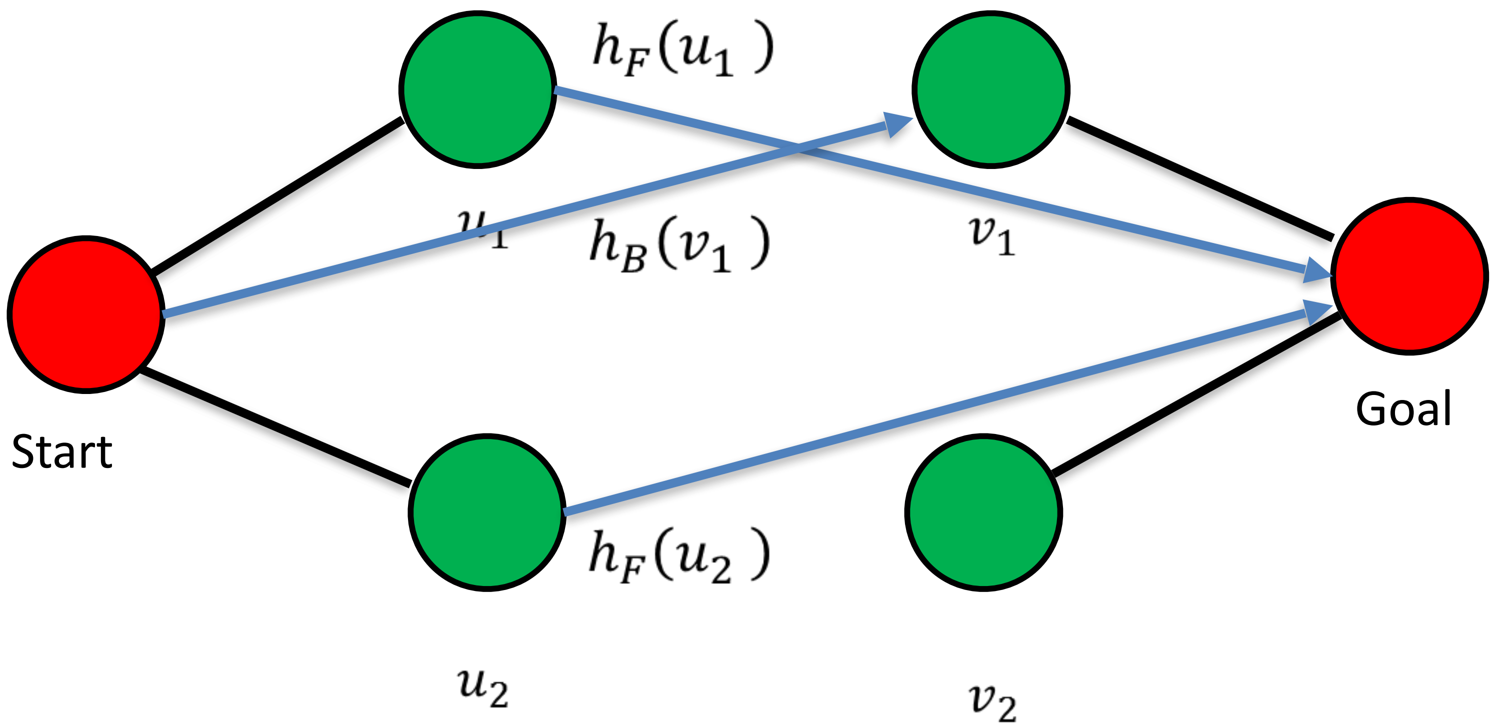
Front-to-End

- Two front-to-end heuristics



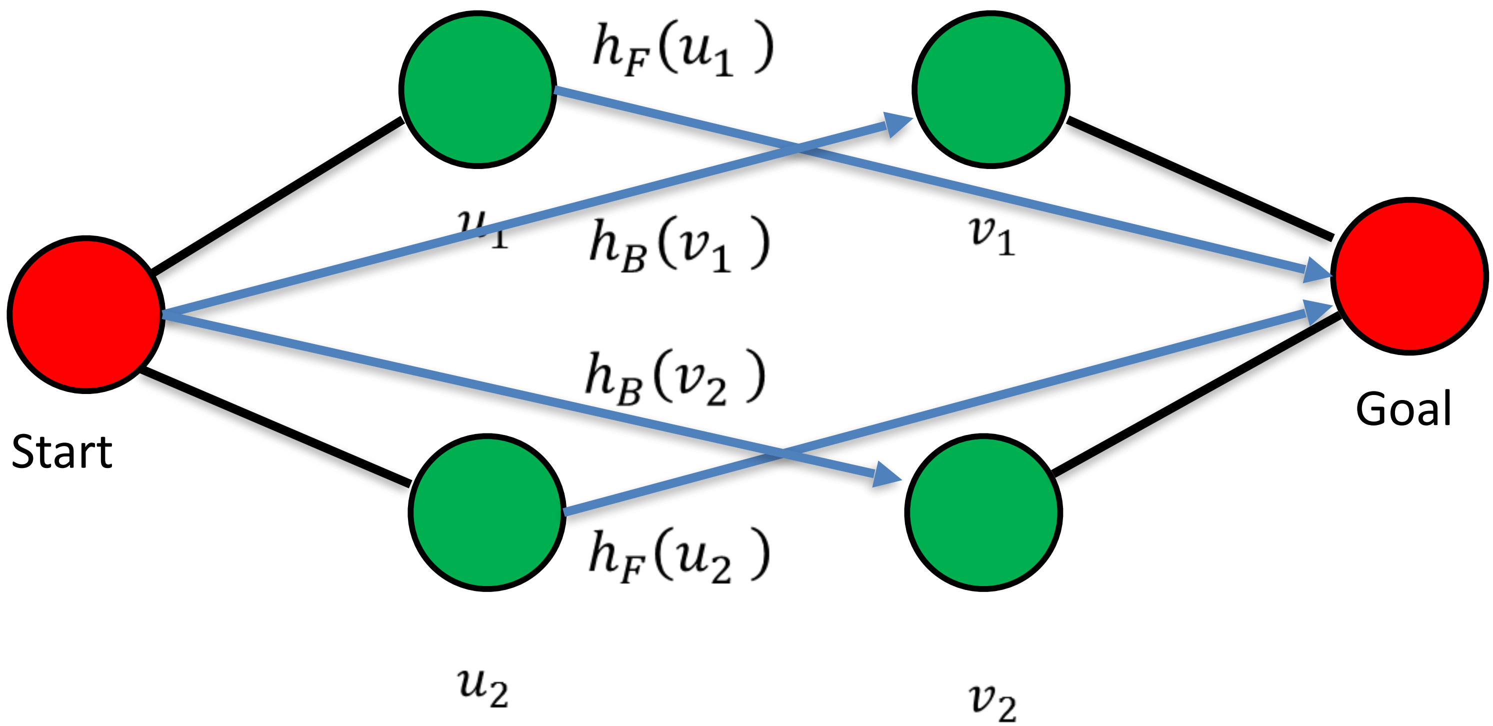
Front-to-End

- Two front-to-end heuristics



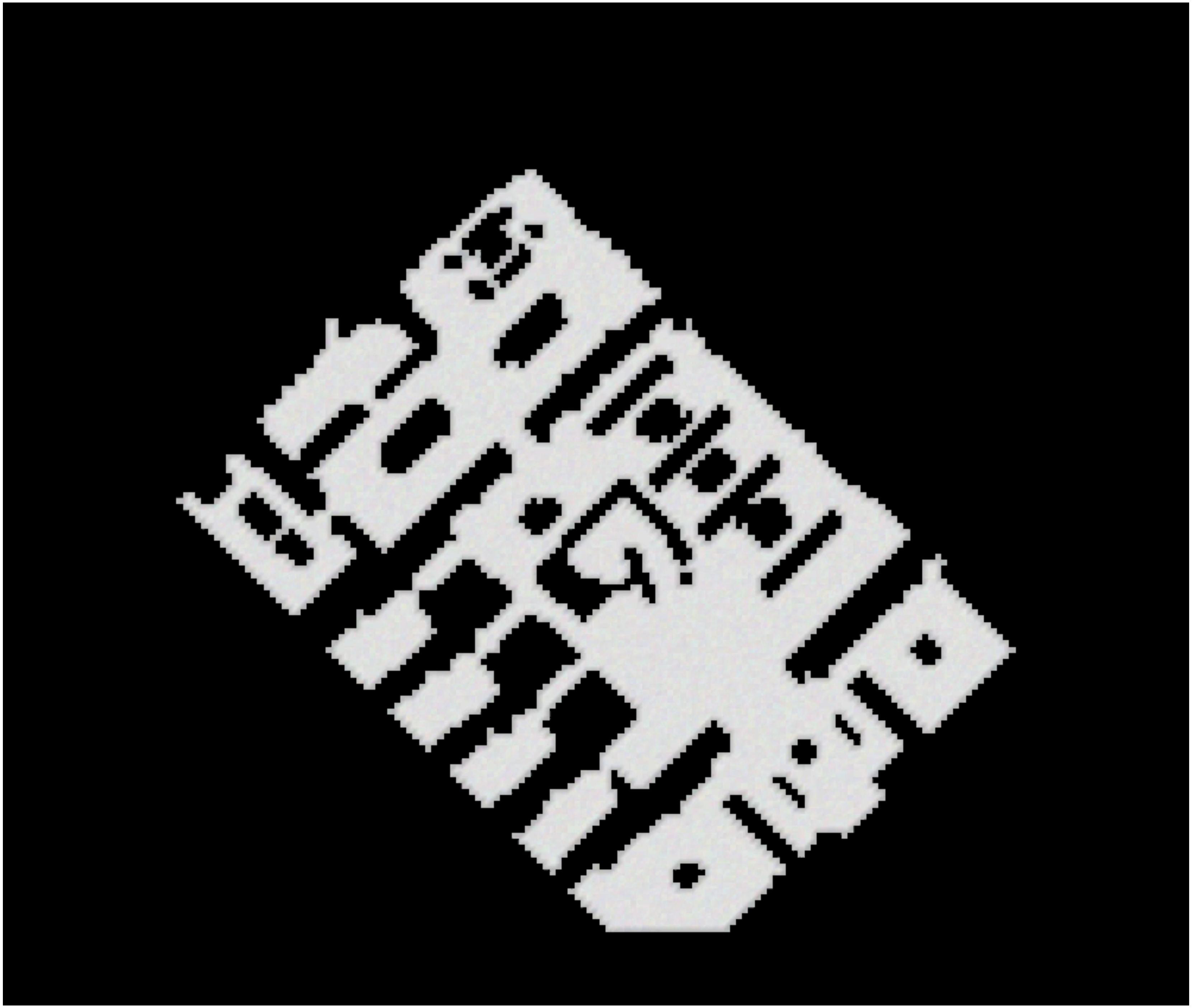
Front-to-End

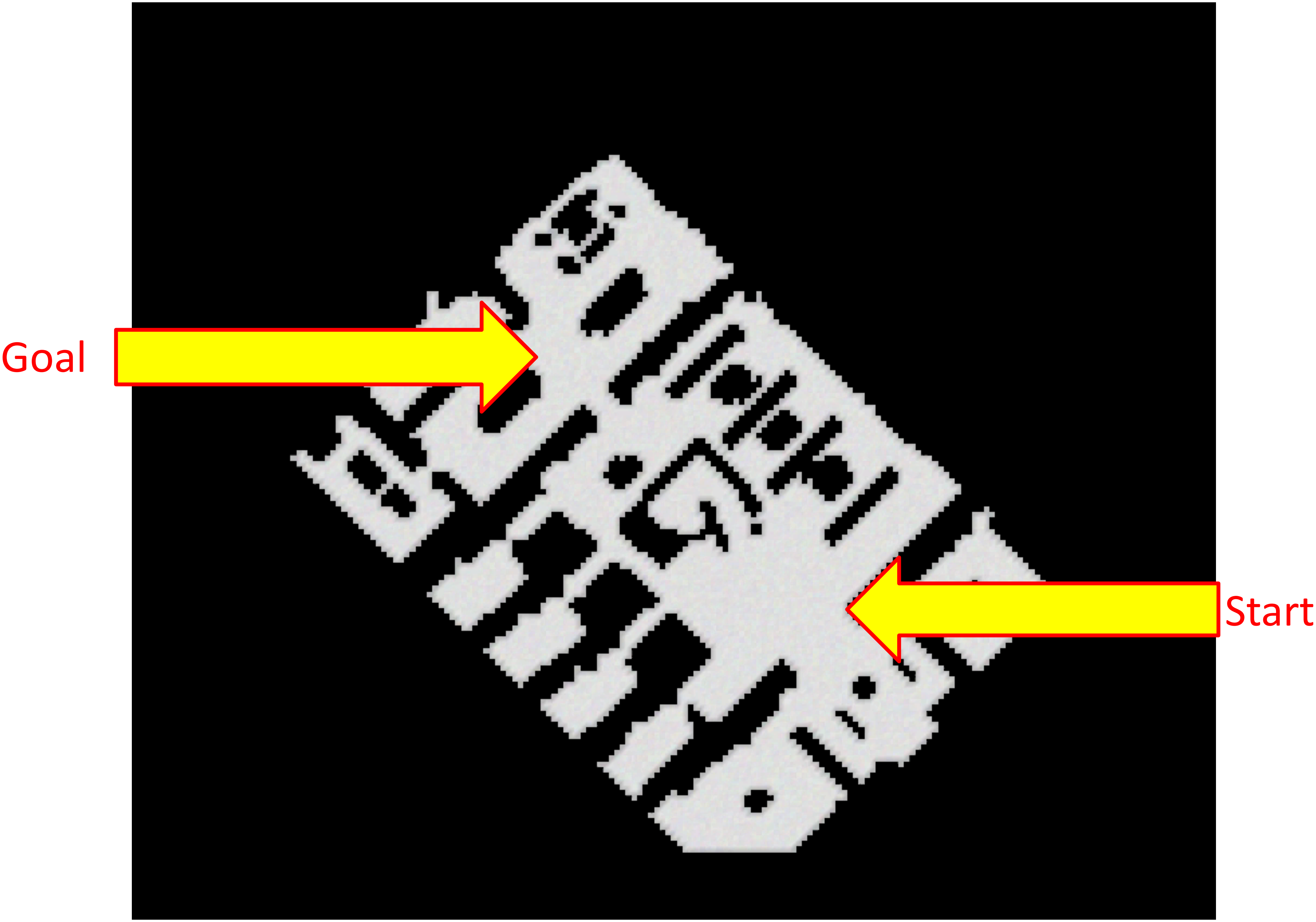
- Two front-to-end heuristics

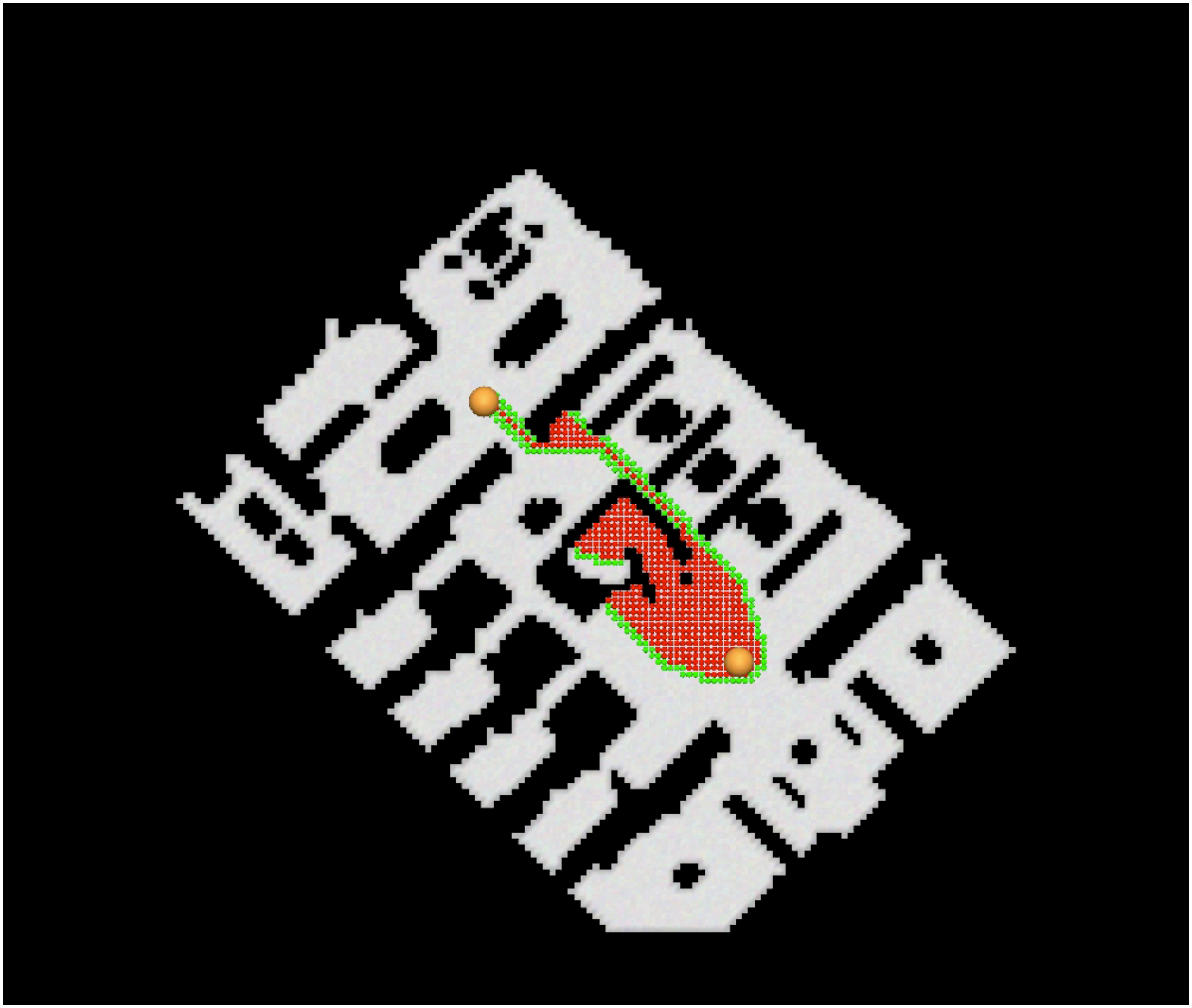


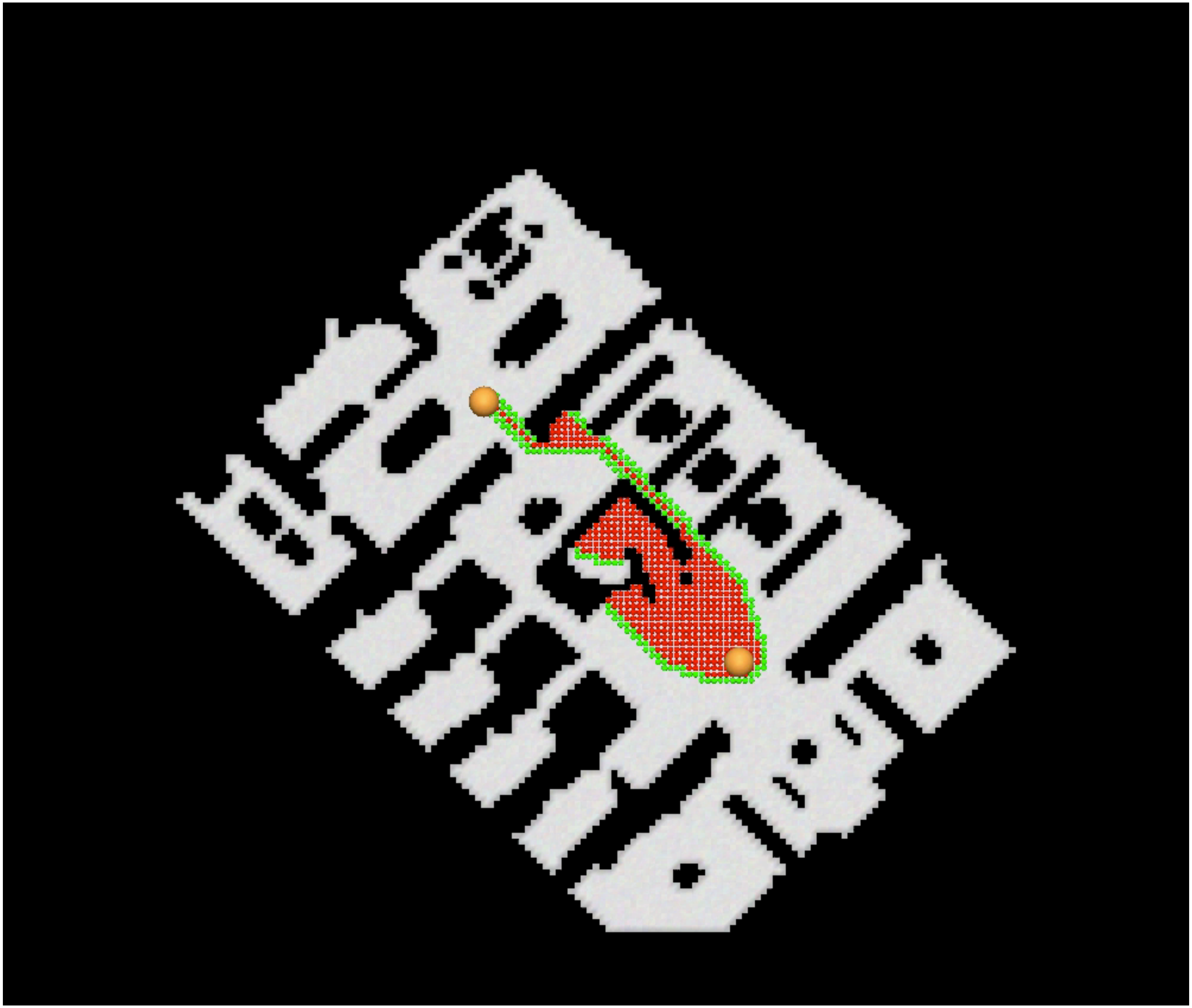
Agenda

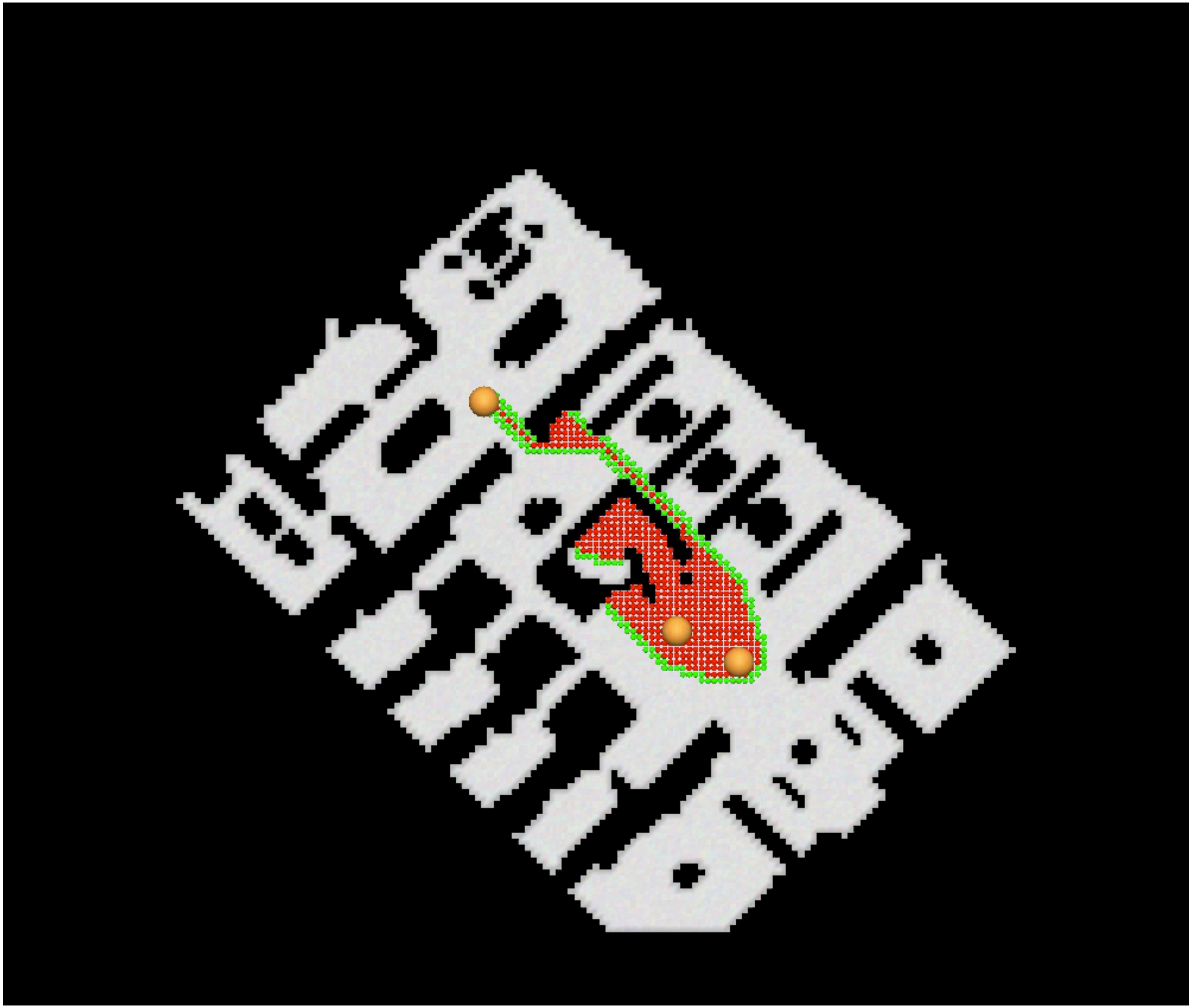
- Assumptions
- Background
  - Unidirectional Search
  - Bidirectional Search
- Main Result
  - No single state must be expanded
  - Sufficient condition
- Applications and Conclusion

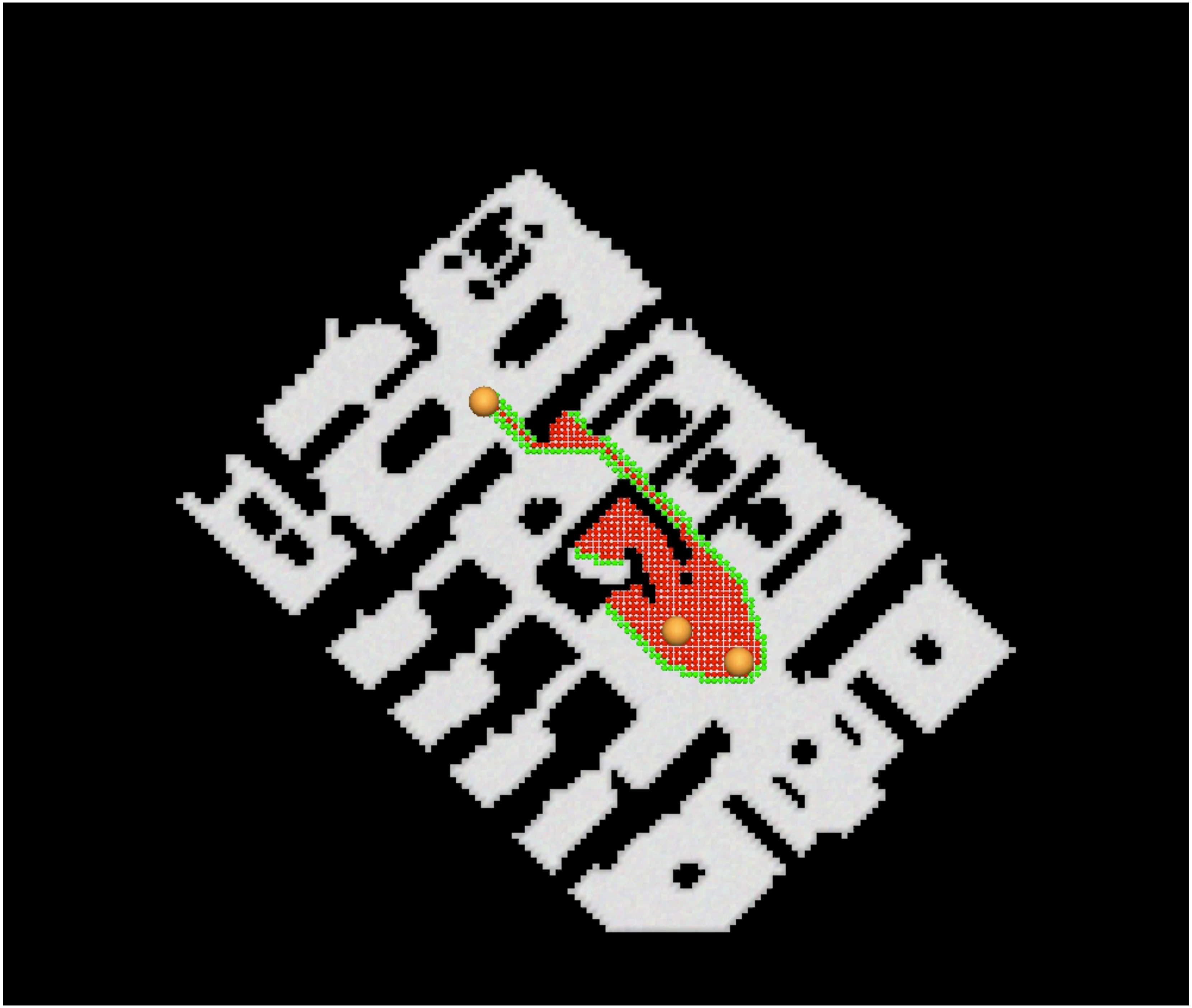


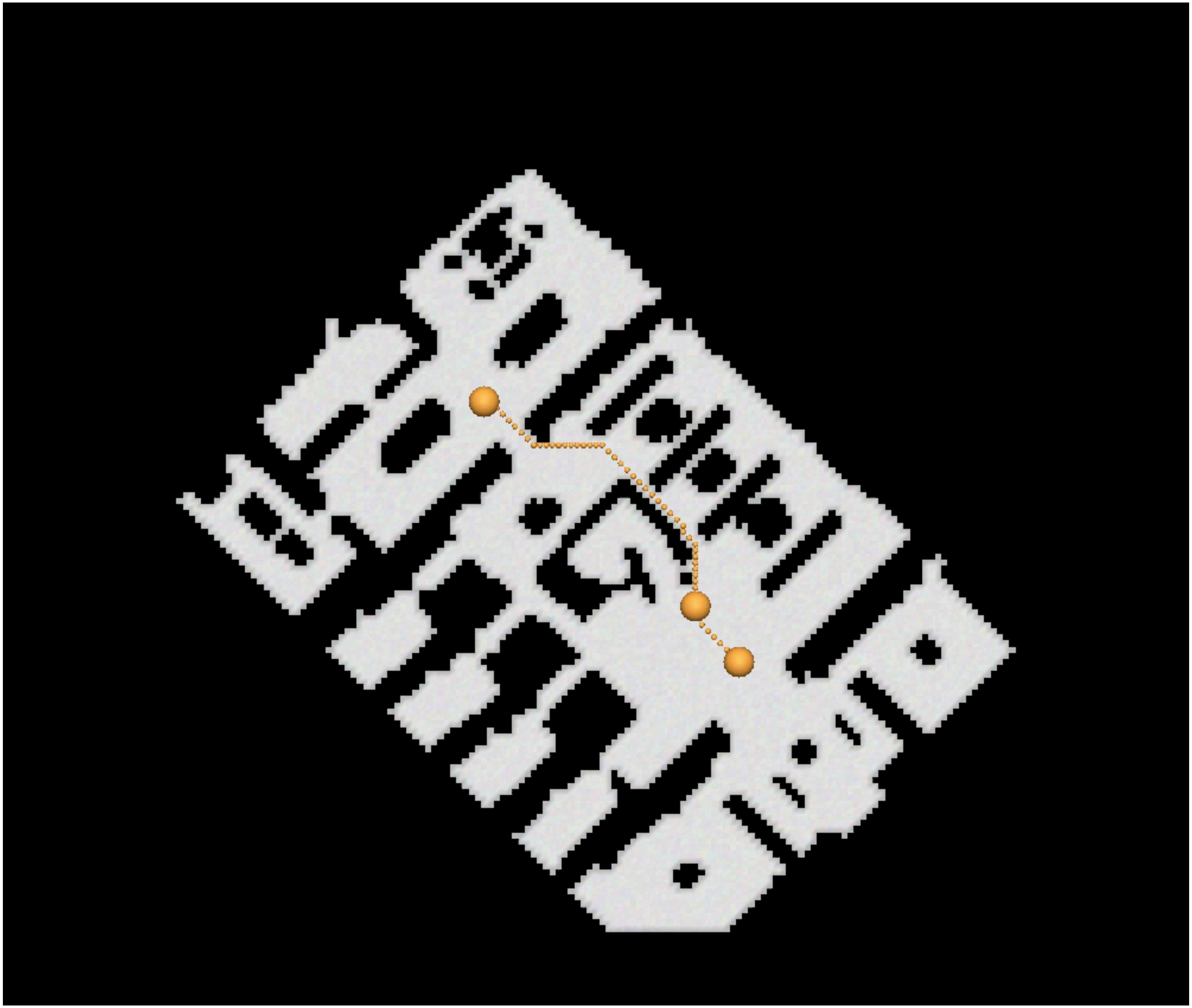

Goal
Start

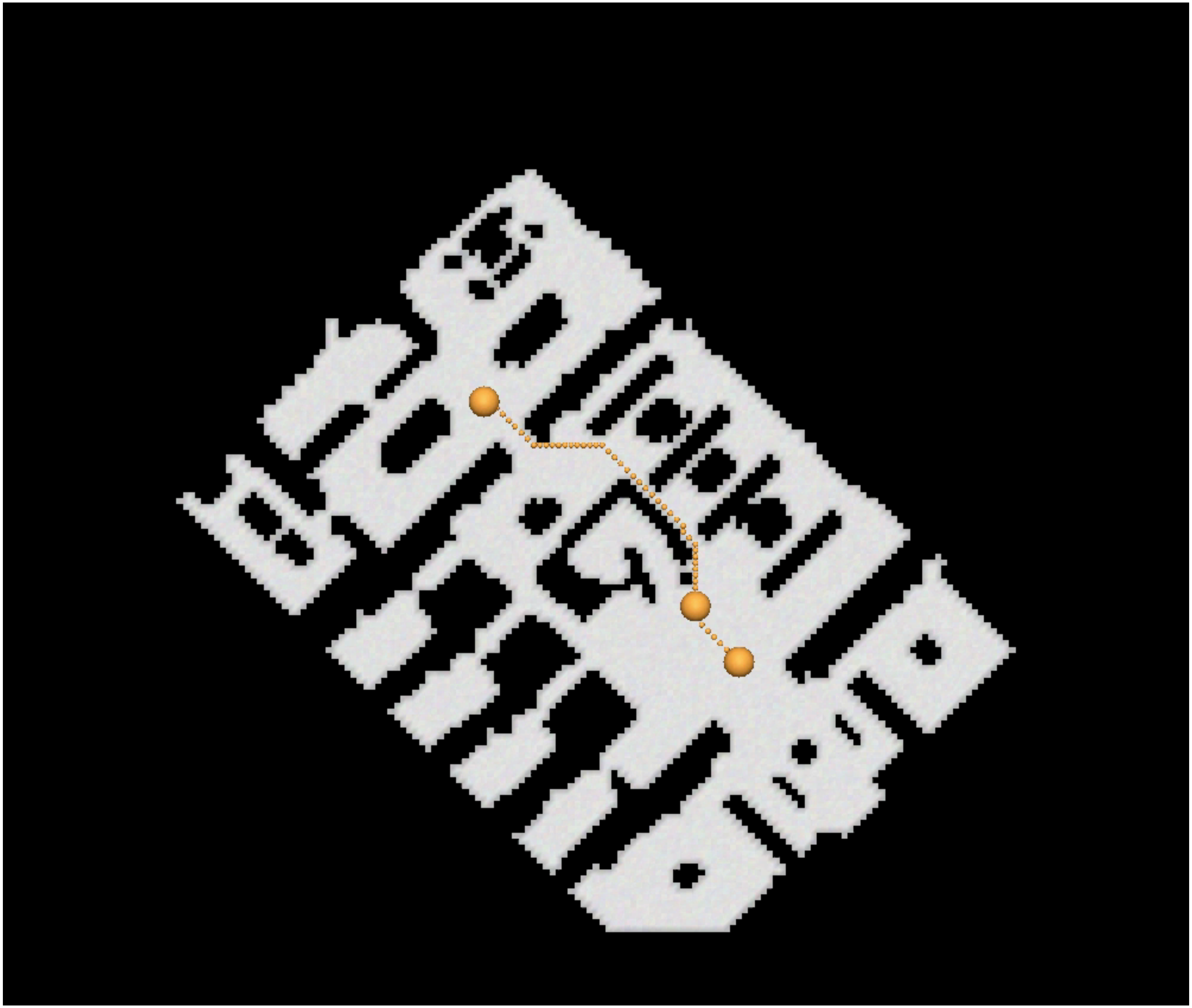


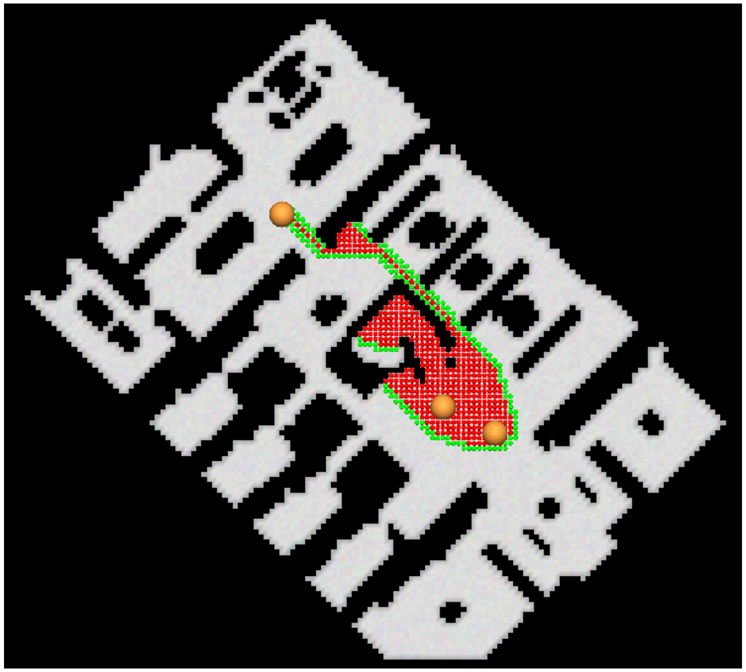




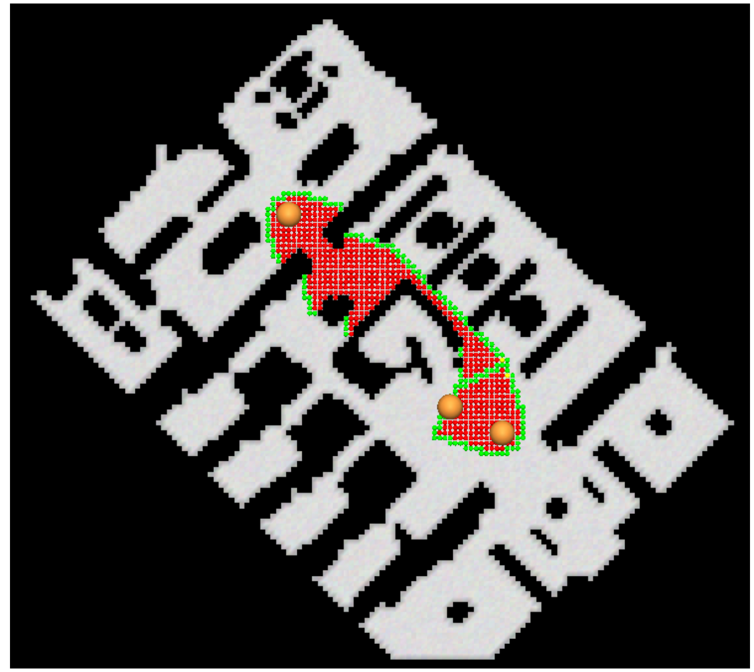




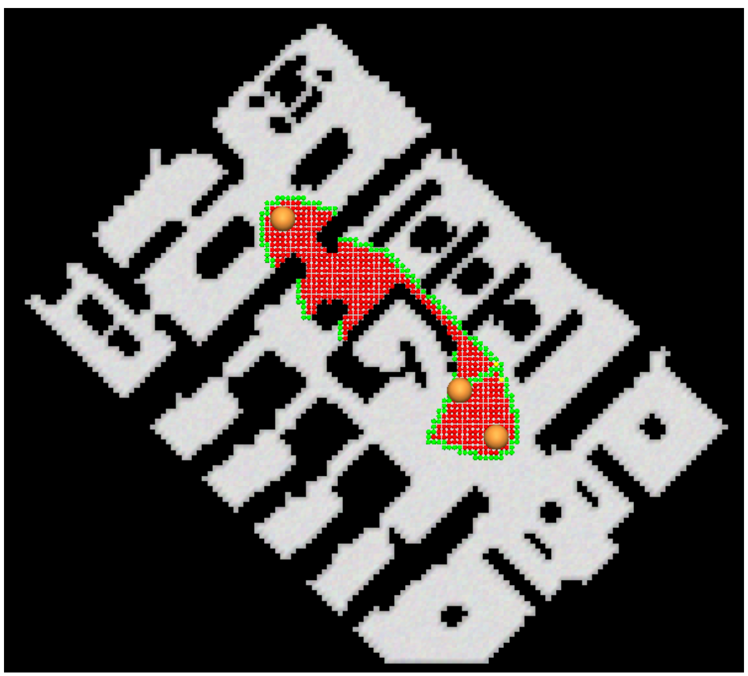




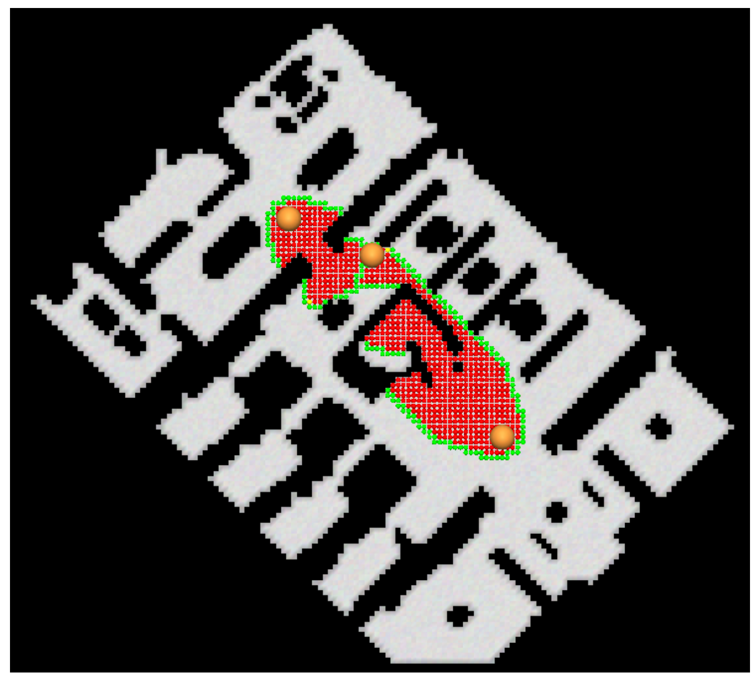
A*



Bidir Alg 1



Bidir Alg 2

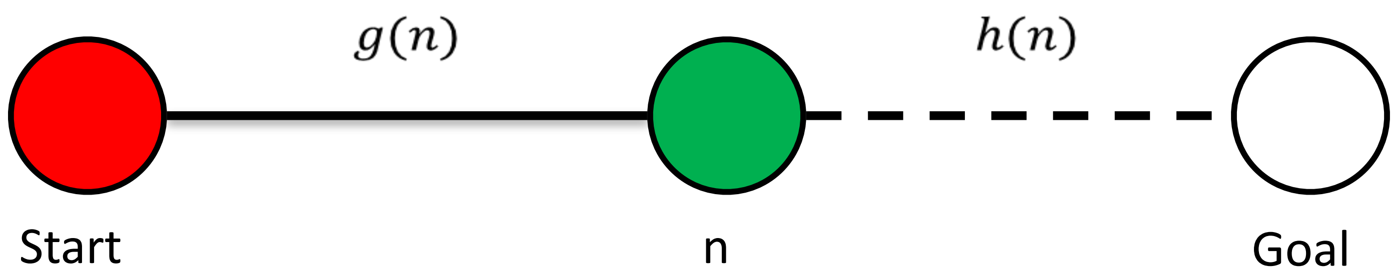


Bidir Alg 3

Key Observation

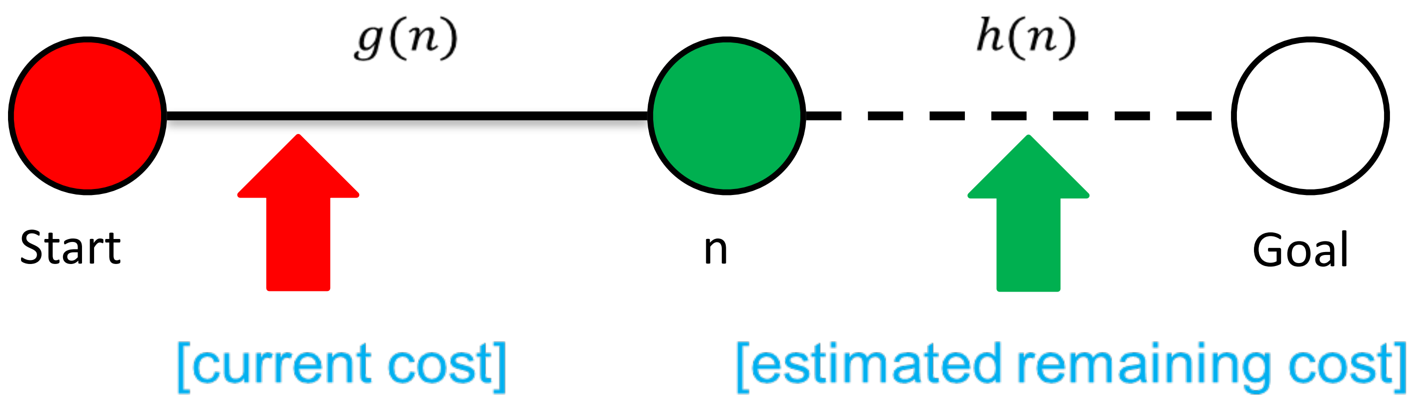
- Bad news: No single state must be expanded!
- Good news: For a pair of states, if they satisfy certain condition (sufficient condition), then at least one of them must be expanded

$f(u, v)$ in front-to-front search

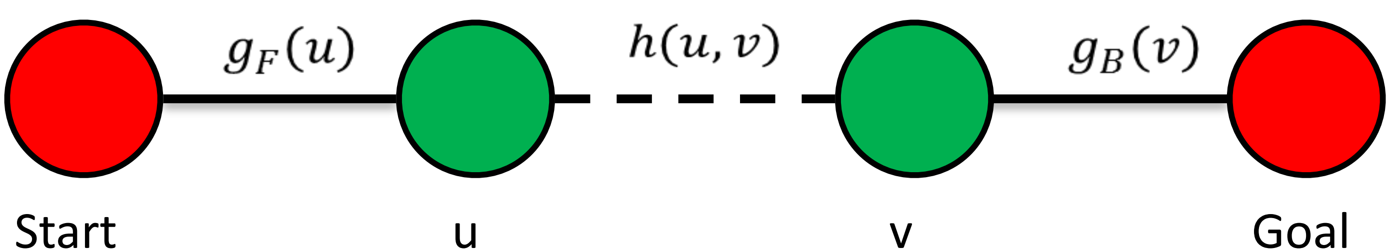


$f(u,v)$ in front-to-front search

Unidirectional:



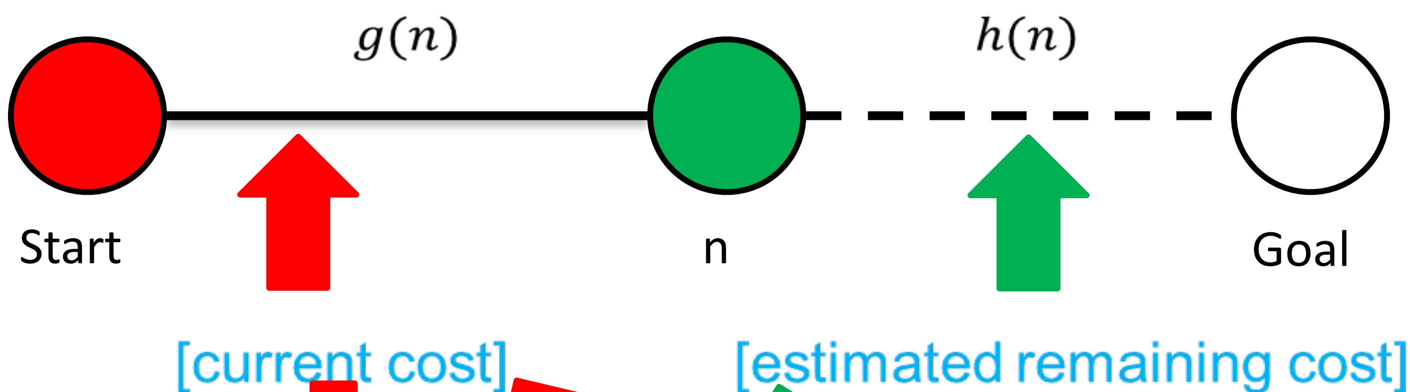
Front-to-front:



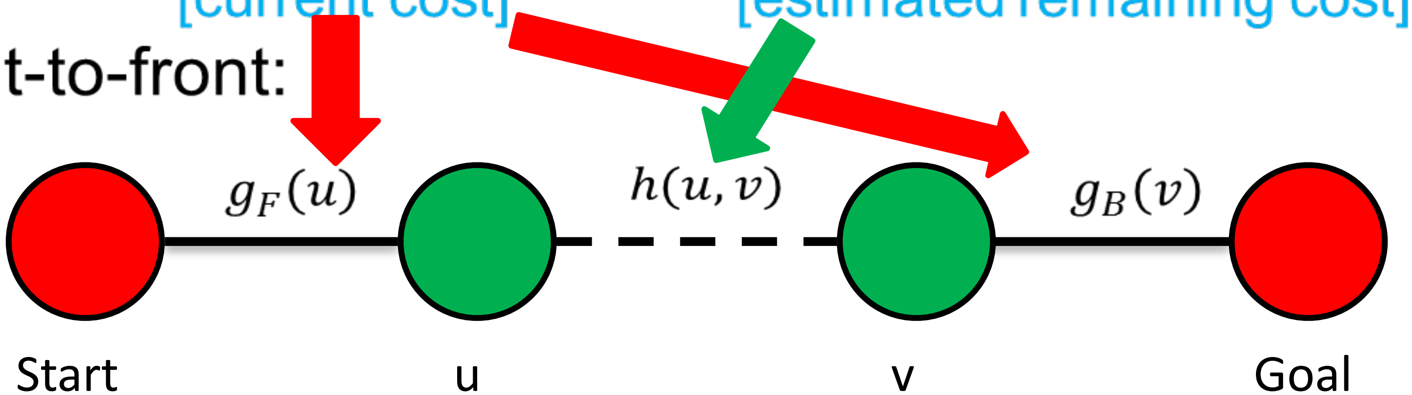
$$f(u,v) \quad = g_F(u) + g_B(v) + h(u,v)$$

$f(u,v)$ in front-to-front search

Unidirectional:



Front-to-front:

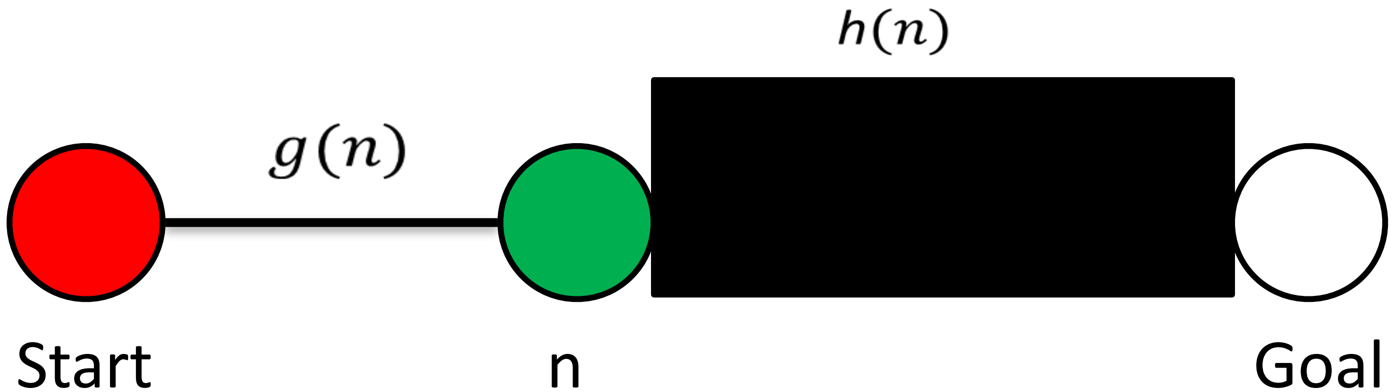


$$f(u,v) \quad = g_F(u) + g_B(v) + h(u,v)$$

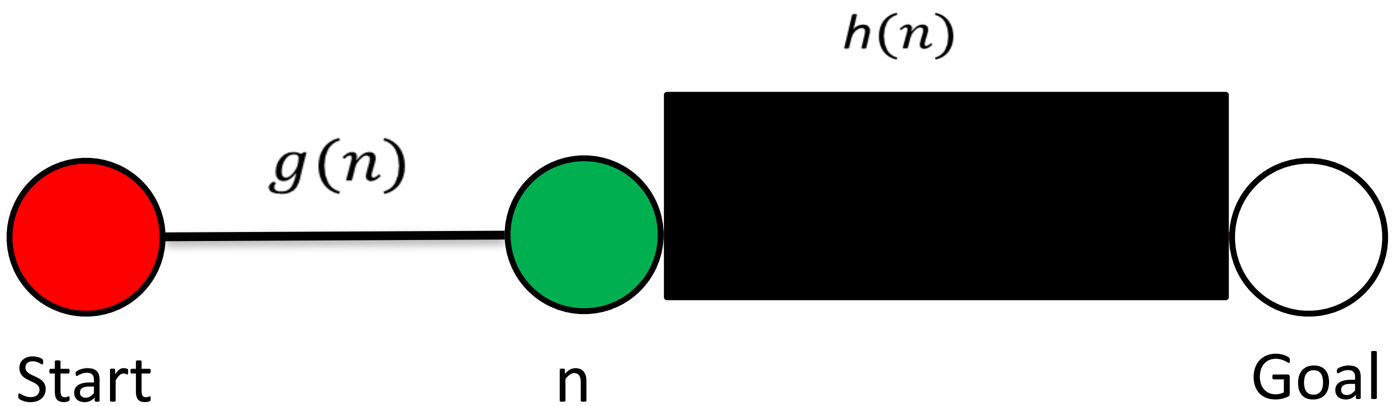
Sufficient Condition

- $f(n) = g(n) + h(n) < C^*$

Instance 1:



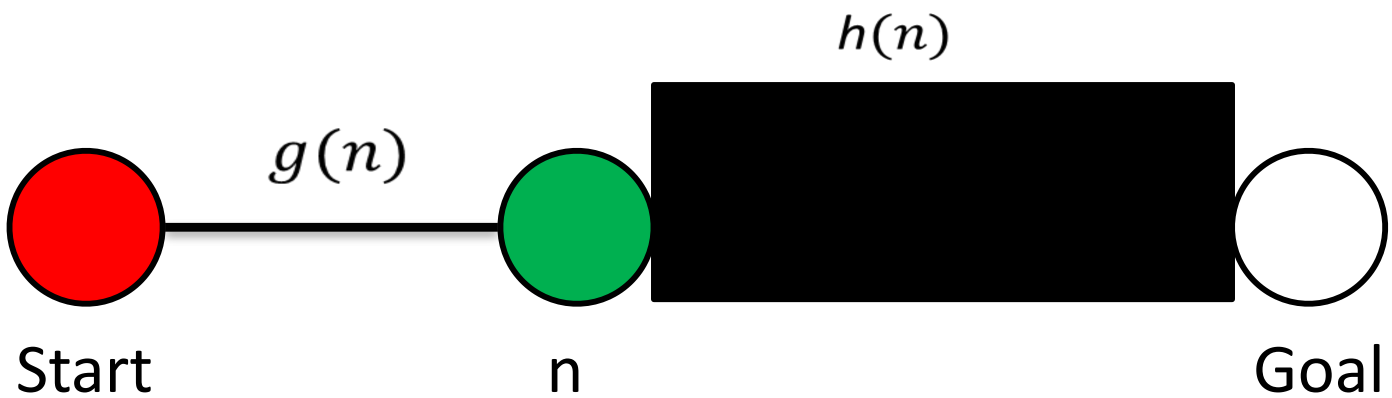
Instance 2:



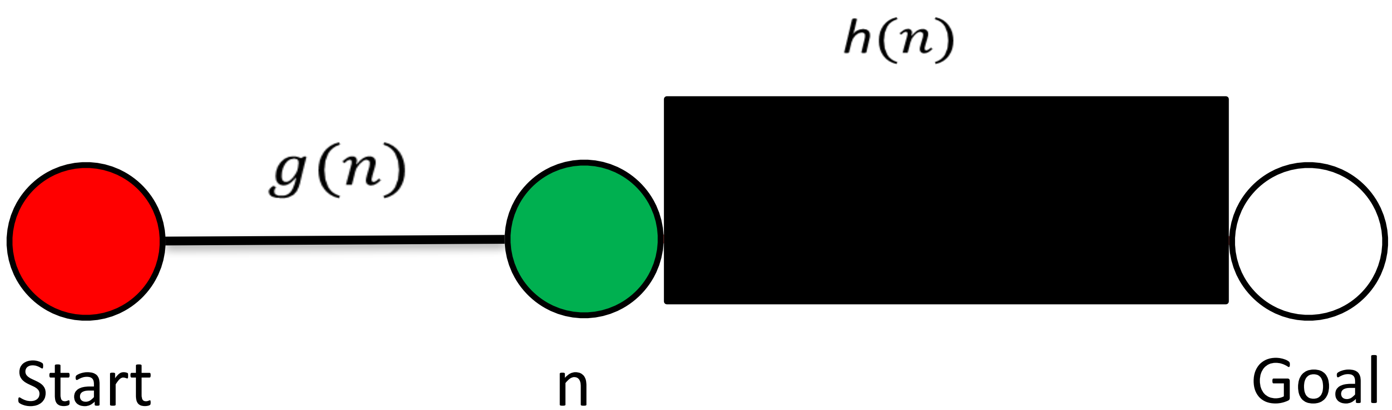
Sufficient Condition

- $f(n) = g(n) + h(n) \; < C^*$

Instance 1:



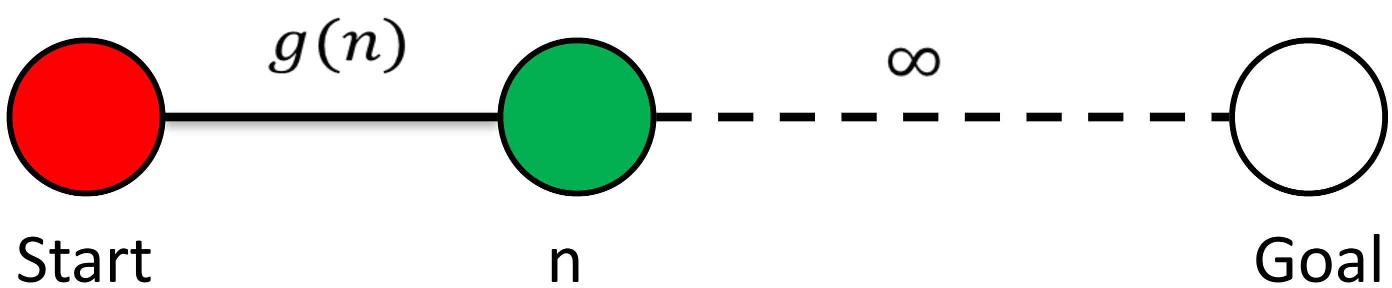
Instance 2:



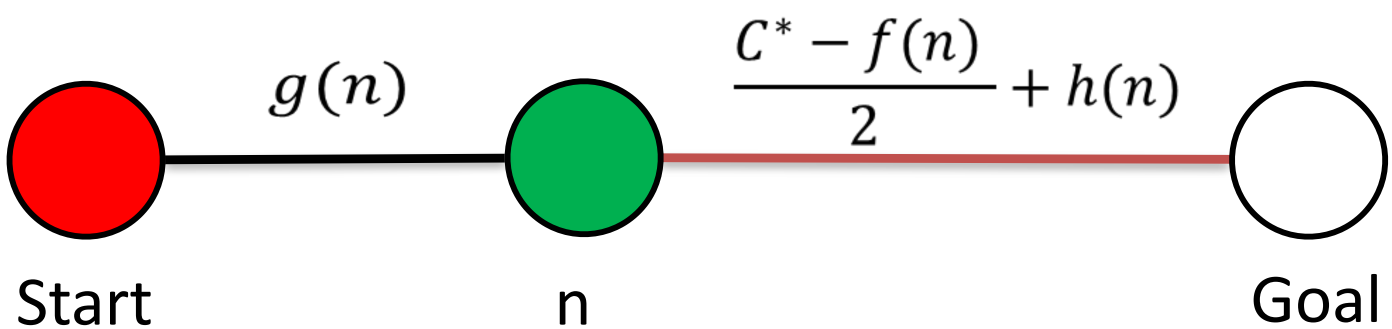
Sufficient Condition

- $f(n) = g(n) + h(n) \; < C^*$

Instance 1:

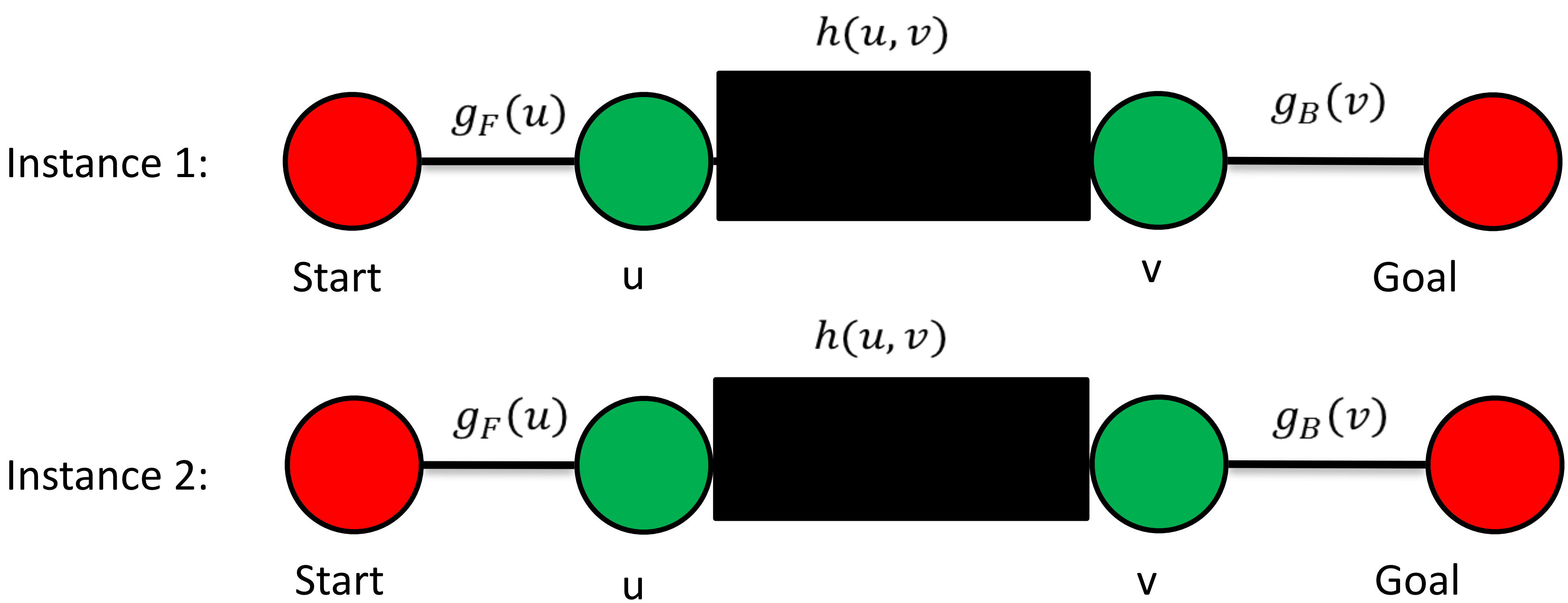


Instance 2:



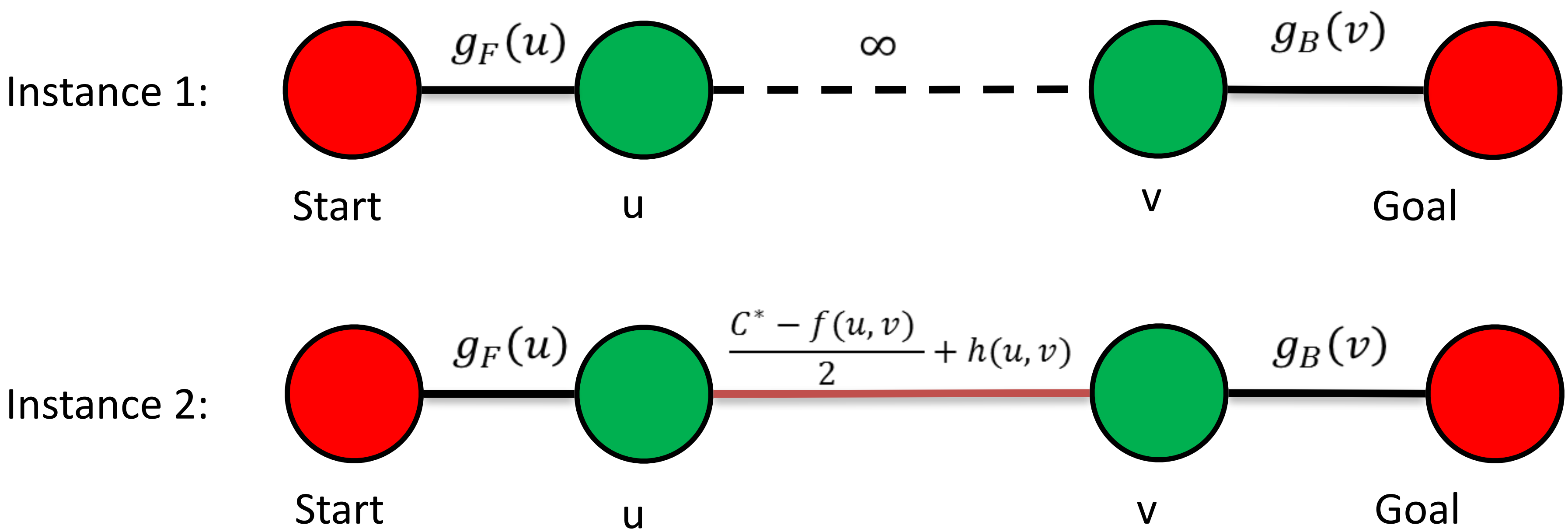
Sufficient Condition

- $f(u, v) < C^*$



Sufficient Condition

- $f(u, v) < C^*$



Sufficient Condition

- Sufficient condition:
  - At least one of (u, v) must be expanded if $f(u, v) < C^*$

Sufficient Condition

- Define $lb(u, v)$ for front-to-end algorithms

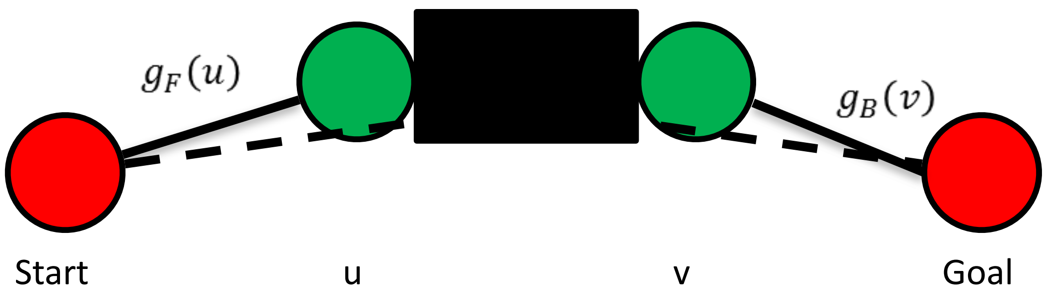
$$lb(u, v) = max \begin{cases} f_F(u) \\ f_B(v) \\ g_F(u) + g_B(v) \end{cases}$$

Sufficient Condition

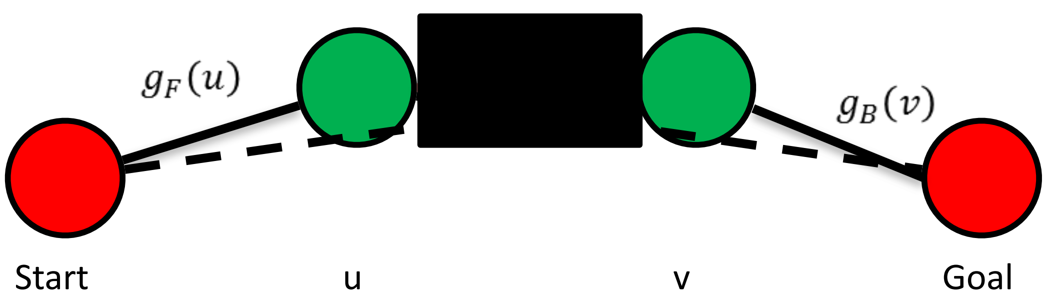
- Front-to-end algorithms:

  $lb(u, v)$<C*

Instance 1:



Instance 2:

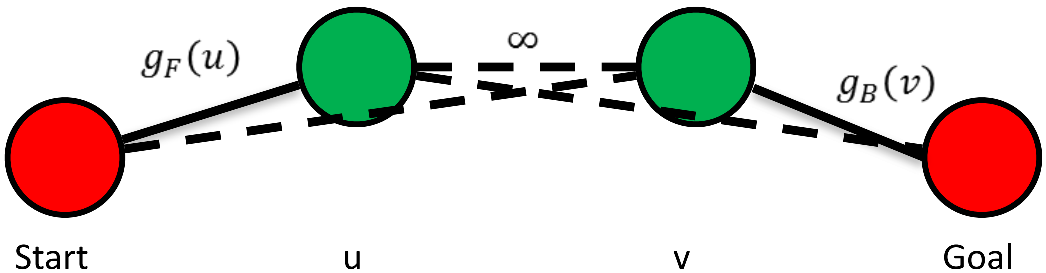


Sufficient Condition

- Front-to-end algorithms:

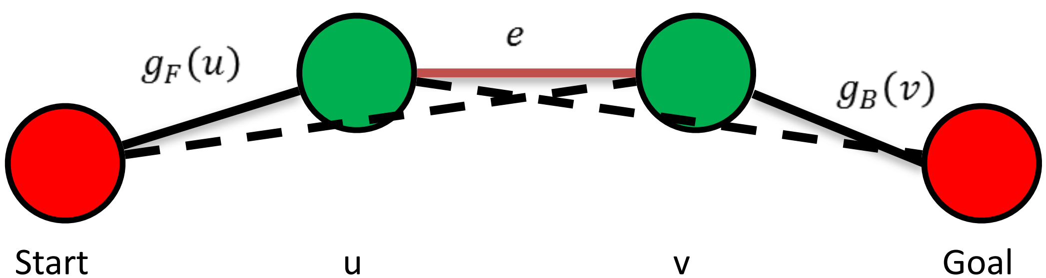
  $lb(u, v)$<C*

Instance 1:



$$e = max \begin{cases} h_F(u) - g_B(v) \\ h_B(v) - g_F(u) \\ \dfrac{C^* - g_F(u) - g_B(v)}{2} \end{cases}$$

Instance 2:



Sufficient Condition

- Sufficient condition:
  - At least one of (u, v) must be expanded if $lb(u, v) < C^*$

Sufficient Condition

- Consider pair (n, goal) where $f(n) < C^*$
  - $f_F(n) < C^*$
  - $f_B(goal) < C^*$
  - $g_F(n) + g_B(goal) = g_F(n) < C^*$

$$lb(n, goal) = max \begin{cases} f_F(n) \\ f_B(goal) \\ g_F(u) + g_B(v) \end{cases} < C^*$$

Agenda

- Assumptions
- Background
  - Unidirectional Search
  - Bidirectional Search
- Main Result
  - No single state must be expanded
  - Sufficient condition
- Applications and Conclusion

Applications

- Post-hoc analysis: minimum node expansions for bidirectional algorithms. [SoCS 2017]
- New front-to-end algorithm: Near-Optimal Bidirectional Search (NBS). [IJCAI 2017]
  - Admissible algorithm
  - Guaranteed bound: 2x in necessary node expansions
  - Tight bound

Conclusion

- Unidirectional algorithms: (1985)
  - n must be expanded if $f(n) < C^*$
- Bidirectional algorithms: (2017)
  - At least one of u or v must be expanded
    - if $f(u, v) < C^*$ (front-to-front)
    - if $lb(u, v) < C^*$ (front-to-end)
  - Unidirectional sufficient condition becomes a special case of front-to-end sufficient condition

Thank you!