Sufficient Conditions for Node Expansion in Bidirectional Heuristic Search

Jürgen Eckerle\textsuperscript{1}, Jingwei Chen\textsuperscript{2}, Nathan R. Sturtevant\textsuperscript{2}, Sandra Zilles\textsuperscript{3} and Robert C. Holte\textsuperscript{4}

\textsuperscript{1}Berner Fachhochschule
\textsuperscript{2}University of Denver
\textsuperscript{3}University of Regina
\textsuperscript{4}University of Alberta

Grant 1551406
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:

- \( g(n) \) between Start and \( n \)
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*

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

Instance 1:

- Start
- $g(n)$
- $\infty$
- Goal

Instance 2:

- Start
- $g(n)$
- $h(n)$
- Goal
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]
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

\[ h(u_1, v_1) \]
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

![Diagram showing two heuristics, $h_F(u_1)$, $h_B(v_1)$, $h_B(v_2)$, and $h_F(u_2)$, connecting start and goal nodes.]
Agenda

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

$$f(u, v) = g_F(u) + g_B(v) + h(u, v)$$
\[ f(u, v) \text{ in front-to-front search} \]

Unidirectional:

\[ g(n) \]

Start \hspace{2cm} n \hspace{2cm} \text{Goal} \hspace{2cm} h(n) \]

Front-to-front:

\[ g_F(u) \]

Start \hspace{2cm} u \hspace{2cm} v \hspace{2cm} \text{Goal} \hspace{2cm} g_B(v) \]

\[ f(u, v) = g_F(u) + g_B(v) + h(u, v) \]
Sufficient Condition

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

• \( f(n) = g(n) + h(n) < C^* \)
Sufficient Condition

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

Instance 1:
- Start
- \( g(n) \)
- \( \infty \)
- Goal

Instance 2:
- Start
- \( g(n) \)
- \( \frac{C^* - f(n)}{2} + h(n) \)
- Goal
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

$$lb(u, v) = \max \left\{ \begin{array}{c} f_F(u) \\ f_B(v) \\ g_F(u) + g_B(v) \end{array} \right\}$$
Sufficient Condition

• Front-to-end algorithms:

\[ lb(u, v) < C^* \]

Instance 1:

```
Start - u - v - Goal
```

Instance 2:

```
Start - u - v - Goal
```
Sufficient Condition

- Front-to-end algorithms:

\[ lb(u, v) < C^* \]

\[
e = \max \left\{ \frac{h_F(u) - g_B(v)}{2}, \frac{h_B(v) - g_F(u)}{2}, C^* - g_F(u) - g_B(v) \right\}
\]
Sufficient Condition

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

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

\[
lb(n, \text{goal}) = \max \left\{ \begin{array}{ll}
    f_F(n) & < C^* \\
    f_B(\text{goal}) & < C^* \\
    g_F(u) + g_B(v) & \end{array} \right.
\]
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!