

Grundlagen der Künstlichen Intelligenz

15. Klassische Suche: A*: Optimalität, Teil I

Malte Helmert

Universität Basel

4. April 2014

Klassische Suche: Überblick

Kapitelüberblick klassische Suche:

- 3.–5. Einführung
- 6.–9. Basisalgorithmen
- 10.–17. heuristische Algorithmen
 - 10. Heuristiken
 - 11. Analyse von Heuristiken
 - 12. Bestensuche als Graphensuche
 - 13. Gierige Bestensuche, A*, Weighted A*
 - 14. IDA*
 - 15. A*: Optimalität, Teil I
 - 16. A*: Optimalität, Teil II
 - 17. A*: Vollständigkeit und Komplexität

Einführung

Optimalität von A*

- Vorteil von A* gegenüber gieriger Suche:
optimal für Heuristiken mit geeigneten Eigenschaften
- **sehr wichtiges Ergebnis!**

~~ die nächsten Kapitel: ein genauerer Blick auf A*

In diesem Kapitel beweisen wir zunächst zwei wichtige Lemmas.

Monotonielemma

A*: Monotonielemma (1)

Lemma (Monotonie von A* mit konsistenten Heuristiken)

Betrachte A* mit einer **konsistenten** Heuristik.

Dann gilt:

- ① Wenn n' ein Kindknoten von n ist, gilt $f(n') \geq f(n)$.
- ② Auf allen von A* erzeugten Pfaden steigen die f -Werte monoton.
- ③ Die Folge der f -Werte der von A* expandierten Knoten steigt monoton.

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

- Definition von f : $f(n) = g(n) + h(s)$, $f(n') = g(n') + h(s')$

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

- Definition von f : $f(n) = g(n) + h(s)$, $f(n') = g(n') + h(s')$
- Definition von g : $g(n') = g(n) + \text{cost}(a)$

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

- Definition von f : $f(n) = g(n) + h(s)$, $f(n') = g(n') + h(s')$
- Definition von g : $g(n') = g(n) + \text{cost}(a)$
- Konsistenz von h : $h(s) \leq \text{cost}(a) + h(s')$

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

- Definition von f : $f(n) = g(n) + h(s)$, $f(n') = g(n') + h(s')$
 - Definition von g : $g(n') = g(n) + \text{cost}(a)$
 - Konsistenz von h : $h(s) \leq \text{cost}(a) + h(s')$
- ~~> $f(n) = g(n) + h(s) \leq g(n) + \text{cost}(a) + h(s')$
 $= g(n') + h(s') = f(n')$

A*: Monotonielemma (2)

Beweis.

zu 1.:

Sei n' ein Kindknoten von n über Aktion a .

Sei $s = n.\text{state}$, $s' = n'.\text{state}$.

- Definition von f : $f(n) = g(n) + h(s)$, $f(n') = g(n') + h(s')$
 - Definition von g : $g(n') = g(n) + \text{cost}(a)$
 - Konsistenz von h : $h(s) \leq \text{cost}(a) + h(s')$
- ~~~ $f(n) = g(n) + h(s) \leq g(n) + \text{cost}(a) + h(s')$
 $= g(n') + h(s') = f(n')$

zu 2.: folgt direkt aus 1.

...

A*: Monotonielemma (3)

Beweis (Fortsetzung).

zu 3.:

- Sei f_b der minimale f -Wert in *open*
am Anfang einer Schleifeniteration in A^* .
Sei n der entfernte Knoten mit $f(n) = f_b$.

A*: Monotonielemma (3)

Beweis (Fortsetzung).

zu 3.:

- Sei f_b der minimale f -Wert in open
am Anfang einer Schleifeniteration in A^* .
Sei n der entfernte Knoten mit $f(n) = f_b$.
- **zu zeigen:** am Ende der Iteration
ist der minimale f -Wert in open mindestens f_b .

A*: Monotonielemma (3)

Beweis (Fortsetzung).

zu 3.:

- Sei f_b der minimale f -Wert in open
am Anfang einer Schleifeniteration in A^* .
Sei n der entfernte Knoten mit $f(n) = f_b$.
- **zu zeigen:** am Ende der Iteration
ist der minimale f -Wert in open mindestens f_b .
- Wir müssen die Operationen betrachten,
die open modifizieren: open.pop_min und open.insert .

A*: Monotonielemma (3)

Beweis (Fortsetzung).

zu 3.:

- Sei f_b der minimale f -Wert in open
am Anfang einer Schleifeniteration in A*.
Sei n der entfernte Knoten mit $f(n) = f_b$.
- **zu zeigen:** am Ende der Iteration
ist der minimale f -Wert in open mindestens f_b .
- Wir müssen die Operationen betrachten,
die open modifizieren: open.pop_min und open.insert .
- open.pop_min kann den minimalen f -Wert in open
nicht verringern (nur erhöhen).

A*: Monotonielemma (3)

Beweis (Fortsetzung).

zu 3.:

- Sei f_b der minimale f -Wert in open
am Anfang einer Schleifeniteration in A*.
Sei n der entfernte Knoten mit $f(n) = f_b$.
- **zu zeigen:** am Ende der Iteration
ist der minimale f -Wert in open mindestens f_b .
- Wir müssen die Operationen betrachten,
die open modifizieren: open.pop_min und open.insert .
- open.pop_min kann den minimalen f -Wert in open
nicht verringern (nur erhöhen).
- Die mit open.insert hinzugefügten Knoten n' sind Kinder
von n und erfüllen daher $f(n') \geq f(n) = f_b$ nach Teil 1.



Optimale-Pfade-Lemma

Optimale Pfade

Zwei weitere Notationen im Zusammenhang mit **optimalen Pfaden** vom Anfangszustand s_0 zu Zuständen s :

- $g^*(s)$: Kosten eines optimalen Pfades von s_0 zu s
- $f_h^*(s) = g^*(s) + h(s)$

Anmerkungen:

- f_h^* und g^* für **Zustände**, nicht Knoten definiert, anders als f und g ([Warum?](#))
- $f_h^*(n.state) \leq f(n)$ für alle Knoten n , die ein Suchalgorithmus erzeugen kann ([Warum?](#))

A*: Optimale-Pfade-Lemma (1)

Lemma (Optimale Pfade für A* mit konsistenter Heuristik)

Sei n ein Knoten, der von A* mit einer **konsistenten Heuristik** expandiert wird, und sei $s = n.state$.

Dann gilt:

- ① $g(n) = g^*(s)$
- ② $f(n) = f_h^*(s)$

In Worten: Wenn A* mit konsistenter Heuristik n expandiert, wurde ein **optimaler Pfad** von s_0 zum Zustand von n gefunden.

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

Induktionsanfang:

Der erste expandierte Knoten ist der Wurzelknoten n_0 für den Anfangszustand s_0 .

Es gilt: $g(n_0) = 0 = g^*(s_0)$ und $f(n_0) = 0 + h(s_0) = f_h^*(s_0)$

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

Induktionsanfang:

Der erste expandierte Knoten ist der Wurzelknoten n_0 für den Anfangszustand s_0 .

Es gilt: $g(n_0) = 0 = g^*(s_0)$ und $f(n_0) = 0 + h(s_0) = f_h^*(s_0)$

Induktionsschritt:

zu 2.:

Betrachte Situation, unmittelbar bevor n von *open* entfernt wird.
Sei $s = n.state$.

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

Induktionsanfang:

Der erste expandierte Knoten ist der Wurzelknoten n_0 für den Anfangszustand s_0 .

Es gilt: $g(n_0) = 0 = g^*(s_0)$ und $f(n_0) = 0 + h(s_0) = f_h^*(s_0)$

Induktionsschritt:

zu 2.:

Betrachte Situation, unmittelbar bevor n von *open* entfernt wird.

Sei $s = n.\text{state}$.

Wir können annehmen, dass n kein Duplikat ist ($s \notin \text{closed}$), denn sonst wird n nicht expandiert.

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

Induktionsanfang:

Der erste expandierte Knoten ist der Wurzelknoten n_0 für den Anfangszustand s_0 .

Es gilt: $g(n_0) = 0 = g^*(s_0)$ und $f(n_0) = 0 + h(s_0) = f_h^*(s_0)$

Induktionsschritt:

zu 2.:

Betrachte Situation, unmittelbar bevor n von *open* entfernt wird.

Sei $s = n.state$.

Wir können annehmen, dass n kein Duplikat ist ($s \notin closed$), denn sonst wird n nicht expandiert.

- Sei s_0, s_1, \dots, s_k mit $s_k = s$ ein **optimaler** Pfad von s_0 zu s .

A*: Optimale-Pfade-Lemma (2)

Beweis.

Induktion über die Anzahl expandierter Knoten:

Induktionsanfang:

Der erste expandierte Knoten ist der Wurzelknoten n_0 für den Anfangszustand s_0 .

Es gilt: $g(n_0) = 0 = g^*(s_0)$ und $f(n_0) = 0 + h(s_0) = f_h^*(s_0)$

Induktionsschritt:

zu 2.:

Betrachte Situation, unmittelbar bevor n von *open* entfernt wird.

Sei $s = n.state$.

Wir können annehmen, dass n kein Duplikat ist ($s \notin closed$), denn sonst wird n nicht expandiert.

- Sei s_0, s_1, \dots, s_k mit $s_k = s$ ein **optimaler** Pfad von s_0 zu s .
- Sei j der grösste Index mit $s_j \in closed$ und $s_{j+1} \notin closed$

A*: Optimale-Pfade-Lemma (3)

Beweis (Fortsetzung).

- So ein Index j existiert immer, da
 - $s_0 \in closed$ (Wurzelknoten schon expandiert)
 - $s_k \notin closed$ (sonst n Duplikat)

A*: Optimale-Pfade-Lemma (3)

Beweis (Fortsetzung).

- So ein Index j existiert immer, da
 - $s_0 \in closed$ (Wurzelknoten schon expandiert)
 - $s_k \notin closed$ (sonst n Duplikat)
- Da $s_j \in closed$ wurde ein Knoten n_j mit $n_j.state = s_j$ expandiert.

A*: Optimale-Pfade-Lemma (3)

Beweis (Fortsetzung).

- So ein Index j existiert immer, da
 - $s_0 \in closed$ (Wurzelknoten schon expandiert)
 - $s_k \notin closed$ (sonst n Duplikat)
- Da $s_j \in closed$ wurde ein Knoten n_j mit $n_j.state = s_j$ expandiert.
- Nach Induktionsvoraussetzung gilt $g(n_j) = g^*(s_j)$.

A*: Optimale-Pfade-Lemma (3)

Beweis (Fortsetzung).

- So ein Index j existiert immer, da
 - $s_0 \in closed$ (Wurzelknoten schon expandiert)
 - $s_k \notin closed$ (sonst n Duplikat)
- Da $s_j \in closed$ wurde ein Knoten n_j mit $n_j.state = s_j$ expandiert.
- Nach Induktionsvoraussetzung gilt $g(n_j) = g^*(s_j)$.
- Daher können wir o.B.d.A. annehmen, dass die Zustände s_0, \dots, s_j so gewählt wurden, dass sie den Pfad zu n_j bilden.
Dies beeinflusst nicht die Optimalität des Pfades s_0, \dots, s_k .

A*: Optimale-Pfade-Lemma (3)

Beweis (Fortsetzung).

- So ein Index j existiert immer, da
 - $s_0 \in closed$ (Wurzelknoten schon expandiert)
 - $s_k \notin closed$ (sonst n Duplikat)
- Da $s_j \in closed$ wurde ein Knoten n_j mit $n_j.state = s_j$ expandiert.
- Nach Induktionsvoraussetzung gilt $g(n_j) = g^*(s_j)$.
- Daher können wir o.B.d.A. annehmen,
dass die Zustände s_0, \dots, s_j so gewählt wurden,
dass sie den Pfad zu n_j bilden.
Dies beeinflusst nicht die Optimalität des Pfades s_0, \dots, s_k .
- Seien n_0, \dots, n_k die Knoten, die zu $\langle s_0, \dots, s_k \rangle$ gehören.
(Es ist nicht nötig, dass diese von A* erzeugt wurden.) ...

A*: Optimale-Pfade-Lemma (4)

Beweis (Fortsetzung).

- Aus $s_j \in closed$ und $s_{j+1} \notin closed$ erhalten wir $n_{j+1} \in open$:
 - Bei der Expansion von n_j wurde n_{j+1} in $open$ eingefügt.
 - Wenn n_{j+1} aus $open$ entnommen worden wäre,
hätten wir $s_{j+1} \in closed$. (Widerspruch!)

A*: Optimale-Pfade-Lemma (4)

Beweis (Fortsetzung).

- Aus $s_j \in closed$ und $s_{j+1} \notin closed$ erhalten wir $n_{j+1} \in open$:
 - Bei der Expansion von n_j wurde n_{j+1} in $open$ eingefügt.
 - Wenn n_{j+1} aus $open$ entnommen worden wäre,
hätten wir $s_{j+1} \in closed$. (Widerspruch!)
- Also sind unmittelbar vor der Expansion von n
sowohl n_{j+1} als auch n in $open$, und n wird ausgewählt.
Daraus folgt $f(n) \leq f(n_{j+1})$.

A*: Optimale-Pfade-Lemma (4)

Beweis (Fortsetzung).

- Aus $s_j \in closed$ und $s_{j+1} \notin closed$ erhalten wir $n_{j+1} \in open$:
 - Bei der Expansion von n_j wurde n_{j+1} in $open$ eingefügt.
 - Wenn n_{j+1} aus $open$ entnommen worden wäre,
hätten wir $s_{j+1} \in closed$. (Widerspruch!)
- Also sind unmittelbar vor der Expansion von n
sowohl n_{j+1} als auch n in $open$, und n wird ausgewählt.
Daraus folgt $f(n) \leq f(n_{j+1})$.
- Wegen der Monotonie von f auf Pfaden:
$$f(n_{j+1}) \leq f(n_{j+2}) \leq \cdots \leq f(n_k)$$

A*: Optimale-Pfade-Lemma (4)

Beweis (Fortsetzung).

- Aus $s_j \in closed$ und $s_{j+1} \notin closed$ erhalten wir $n_{j+1} \in open$:
 - Bei der Expansion von n_j wurde n_{j+1} in $open$ eingefügt.
 - Wenn n_{j+1} aus $open$ entnommen worden wäre,
hätten wir $s_{j+1} \in closed$. (Widerspruch!)
- Also sind unmittelbar vor der Expansion von n
sowohl n_{j+1} als auch n in $open$, und n wird ausgewählt.
Daraus folgt $f(n) \leq f(n_{j+1})$.
- Wegen der Monotonie von f auf Pfaden:
$$f(n_{j+1}) \leq f(n_{j+2}) \leq \cdots \leq f(n_k)$$
- Wegen der Optimalität des Pfades s_0, \dots, s_k :
$$f(n_k) = g(n_k) + h(s_k) = g^*(s_k) + h(s_k) = f_h^*(s_k) = f_h^*(s)$$

A*: Optimale-Pfade-Lemma (4)

Beweis (Fortsetzung).

- Aus $s_j \in closed$ und $s_{j+1} \notin closed$ erhalten wir $n_{j+1} \in open$:
 - Bei der Expansion von n_j wurde n_{j+1} in $open$ eingefügt.
 - Wenn n_{j+1} aus $open$ entnommen worden wäre,
hätten wir $s_{j+1} \in closed$. (Widerspruch!)
- Also sind unmittelbar vor der Expansion von n
sowohl n_{j+1} als auch n in $open$, und n wird ausgewählt.
Daraus folgt $f(n) \leq f(n_{j+1})$.
- Wegen der Monotonie von f auf Pfaden:
$$f(n_{j+1}) \leq f(n_{j+2}) \leq \cdots \leq f(n_k)$$
- Wegen der Optimalität des Pfades s_0, \dots, s_k :
$$f(n_k) = g(n_k) + h(s_k) = g^*(s_k) + h(s_k) = f_h^*(s_k) = f_h^*(s)$$
- Zusammen: $f(n) \leq f_h^*(s)$ und daher $f(n) = f_h^*(s)$
(sonst Widerspruch zur Definition von f_h^*)

...

A*: Optimale-Pfade-Lemma (5)

Beweis (Fortsetzung).

zu 1.:

- aus 2.: $f(n) = f_h^*(s)$
- Definition: $g(n) + h(s) = g^*(s) + h(s)$
- Subtraktion von $h(s)$: $g(n) = g^*(s)$

Im letzten Schritt verwenden wird, dass $h(s)$ endlich sein muss,
falls n expandiert wird.



Zwischenbemerkung: Optimalität von A^{*}?

- Das Ziel unserer Analyse ist, die Optimalität von A^{*} unter geeigneten Voraussetzungen nachzuweisen.
- Reichen unsere bisherigen Ergebnisse hierfür aus?

Zwischenbemerkung: Optimalität von A^{*}?

- Das Ziel unserer Analyse ist, die Optimalität von A^{*} unter geeigneten Voraussetzungen nachzuweisen.
- Reichen unsere bisherigen Ergebnisse hierfür aus?
- Fast!** Das Optimale-Pfade-Lemma garantiert:
Wenn A^{*} einen Zielknoten aus *open* entnimmt, wurde
ein optimaler Pfad zum zugehörigen Zielzustand gefunden.
- Aber vielleicht gibt es billigere Pfade
zu **anderen** Zielzuständen?
- Ohne weitere Forderungen an die Heuristik ist dies
tatsächlich möglich! (Konsistenz alleine reicht nicht.)

Einführung
oo

Monotonielemma
oooo

Optimale-Pfade-Lemma
oooooooo

Zusammenfassung
●○

Zusammenfassung

Zusammenfassung

Beginn Optimalitätsbeweis von A*:

- Monotonie-Lemma für A* mit konsistenter Heuristik:
 - f -Werte steigen monoton von Knoten zu Kindknoten
 - f -Werte steigen monoton auf Pfaden
 - f -Werte der expandierten Knoten steigen monoton
- Optimale-Pfade-Lemma für A* mit konsistenter Heuristik:
Wenn n expandiert wird, wurde ein optimaler Pfad von der Wurzel zum Zustand von n gefunden.