1

Der A^*-Algorithmus

Voraussetzung 1.1 *V ist eine Zustandsmenge mit $|N(v)| < +\infty$ für alle $v \in V$. Implizit wird dadurch der gerichtete Graph $G = (V, R)$ mit $R = \{ (v, w) : w \in N(v) \}$ definiert. Es sei $k\colon R \to \mathbb{R}$ eine Kantenbewertung.*

Gegeben sei ferner eine Startmenge $S \subseteq V$ und eine ausgezeichnete Menge $F \subseteq V$, die sogenannte „Zielmenge“.

Definition 1.2
*Für einen Weg $p = (v_0, v_1, \ldots, v_r)$ in G definieren wir seine **Kosten** $k(p)$ durch*

$$k(p) := \sum_{i=0}^{r-1} k(v_i, v_{i+1}).$$

Für eine Ecke $v \in V$ definieren wir ferner

$$\begin{aligned} g^*(v) &:= \inf\{ k(p) : p \text{ ist Weg von } S \text{ nach } v \} \\ h^*(v) &:= \inf\{ k(p) : p \text{ ist Weg von } v \text{ zu einer Ecke in } F \}, \end{aligned}$$

wobei $\inf \varnothing := +\infty$ sei.

Problem: Finde einen Weg p^* von S nach F mit $k(p^*) = K^* = \inf\{ h^*(s) : s \in S \}$.

Bemerkung 1.3 *Es kann $K^* = -\infty$ gelten, selbst wenn G endlich ist.*

Ferner kann bei unendlichem G zwar $K^* > -\infty$ gelten, aber dennoch kein optimaler Weg existieren, weil zwar die Infima, nicht aber die Minima, existieren.

Definition 1.4 (Optimistischer Schätzer)
*$h\colon V \to \mathbb{R}_{\geq 0}$ ist ein **optimistischer (pessimistischer) Schätzer**, wenn $h(v) \leq h^*(v)$ ($h(v) \geq h^*(v)$) für alle $v \in V$ gilt.*

Algorithmus A^* benutzt:

1. Eine Bewertung $g\colon V \to \mathbb{R}$, wobei $g(v)$ die Länge des bisher kürzesten Weges $p(v)$ aus der Startmenge S zur Ecke v ist.

2. Eine Schätzfunktion h für den Restaufwand.

3. Eine Bewertungsfunktion $\Phi = g + h$, um auszuwählen, welche Ecke als nächstes „expandiert“ wird, d.h. deren Nachfolger zur „Kandidatenmenge“ hinzugefügt werden.

Der A^*-Algorithmus ist in Algorithmus 1.1 dargestellt. Er enthält unter anderem folgende „klassische Algorithmen“ als Spezialfälle:

- Falls $k \geq 0$ und $h \equiv 0$, so erhält man den *Dijkstra-Algorithmus.*
- Falls $k \equiv 1$ und $h \equiv 0$, so erhält man *Breadth-First-Search.*
- Falls $k \equiv -1$ und $h \equiv 0$, so erhält man *Depth-First-Search* (bei Verbieten von Wiederbelebungen!): Die Bewertungsfunktion Φ erfüllt dann: $\Phi(w) = \Phi(v) - 1$ für alle $w \in N(v)$.
- Falls K eine vorgegebene Kostenobergrenze ist, die iterativ erhöht wird und die von f nicht überschritten werden darf, so erhält man den sogenannten *Iterativ-Deepening-Algorithmus* (IDA).

Satz 1.5 *Sei $G = (V, R)$ ein Graph, in dem jede Ecke endlichen Außengrad besitzt, S endlich und $k\colon R \to \mathbb{R}_{\geq 0}$, so daß entweder G endlich ist oder es ein $\varepsilon > 0$ mit $k(r) \geq \varepsilon > 0$ für alle $r \in R$. Ferner gebe es einen kostengünstigsten Weg in G von S nach F (im Falle der Endlichkeit von G ist diese Aussage trivial). Dann terminiert der A^*-Algorithmus nach endlich vielen Schritten mit einem $v \in F$.*

Beweis: Nach Voraussetzung existiert ein kostengünstigster Weg $p^* = (v_0, \ldots, v_r)$ mit $v_0 \in S$ und $v_r \in F$. Seine Kosten seien K^*.

1. Nach dem $a$ten Durchlauf der while-Schleife gilt:

   Zu jeder Ecke $v \in \mathsf{open} \cup \mathsf{closed}$ ist $p[v]$ ein Weg von S zu v der Kosten $g[v]$. Es gilt also stets $g[v] \geq g^*(v)$.

   Beweis:

   Induktion nach a. Für $a = 0$ sind die Aussagen trivial.

   Induktionsschritt $a \to a + 1$:

   Wird in Schritt 18 oder 22 $p[w]$ für ein w auf $p[v] \oplus w$ (neu) gesetzt, so ist nach IV $p[v]$ ein Weg von S zu v der Länge $g[v]$. Die erste Behauptung folgt somit. Aus $g[v] = k(p[v]) \geq g^*(v)$ folgt die zweite.

2. Während des Laufs des Algorithmus gilt für $M := \max\{\, i : v_0, \ldots, v_i \in \mathsf{open} \cup \mathsf{closed} \,\}$, daß $v_M \in \mathsf{open}$.

   Man beachte, daß es ein solches M gibt, da anfangs $v_0 \in \mathsf{open}$ und somit invariant immer $v_0 \in \mathsf{open} \cup \mathsf{closed}$ gilt.

   Wir zeigen $v_M \in \mathsf{open}$, indem wir $v_M \in \mathsf{closed}$ zum Widerspruch führen. Da der Algorithmus noch nicht abgebrochen hat, gilt $M < r$. Nachdem

Algorithmus 1.1 A^*-Algorithmus

Input: Startmenge S, Zielmenge F, Zugübergangskosten $k\colon R \to \mathbb{R}$, Schätzfunktion $h\colon V \to \mathbb{R}$

```
{Bedeutung der Variablen:
p[v]: bisher bester Weg von S zu v
g[v]: Kosten von p[v]}
for all s ∈ S do
  p[s] ← ()   {leerer Weg}
  g[s] ← 0
end for
open ← S
closed ← ∅   {leere Menge}
while open ≠ ∅ do
  Wähle das Element v aus open mit minimalem Wert Φ(v) = g[v] + h(v).
  Entferne v aus open.
  Füge v zu closed hinzu.
  if v ∈ F then   {Ziel ist erreicht}
    return p[v] als Lösung
  else   {"'Expansion"' von v}
    for all w ∈ N(v) do
      if w ∉ open ∧ w ∉ closed then   {Neue Ecke}
        Füge w zu open hinzu.
        p[w] ← p[v] ⊕ w   {Füge w an p[v] an.}
        g[w] ← g[v] + k(v, w)
      else   {Bereits bekannte Ecke}
        if g[v]+k(v, w) < g[w] then   {Besserer Weg als bekannter Weg}
          p[w] ← p[v] ⊕ w
          g[w] ← g[v] + k(v, w)
          if w ∈ closed then
            Entferne w aus closed.
            Füge w zu open hinzu.   {„Wiederbelebung“ von w}
          end if
        end if
      end if
    end for
  end if
end while
return „Es gibt keinen Weg“.
```

v_M in closed aufgenommen wurde, war $v_{M+1} \in \mathsf{open} \cup \mathsf{closed}$. Also gilt auch jetzt $v_{M+1} \in \mathsf{open} \cup \mathsf{closed}$ im Widerspruch zur Wahl von M.

3. Während des Laufs des Algorithmus invariant: $\{v_0, \ldots, v_r\} \cap \mathsf{open} \neq \emptyset$.

4. Mit $m := \min\{\, i : v_i \in \mathsf{open} \,\}$ gilt stets: $g[v_m] = g^*(v_m)$.

   Sei $v_m \in \mathsf{open}$ mit m wie oben definiert. Dann gilt nach Punkt 3: $v_0, \ldots, v_{m-1} \in \mathsf{closed}$. Wir zeigen, $g[v_i] \leq g^*(v_i)$ für $i = 0, \ldots, m$. Da nach Punkt 1 immer $g[v_i] \geq g^*(v_i)$ gilt, folgt dann die Gleicheit $g[v_i] = g^*(v_i)$ für alle i.

   Für $i = 0$ ist die Aussage offensichtlich, da anfangs $g[v_0] = 0 = g^*(v_0)$ ist und g nie erhöht wird. Gelte bereits $g[v_j] \leq g^*(v_j)$ für $j = 0, \ldots, i$. Nachdem v_i das letzte Mal in closed aufgenommen wird, gilt wegen $v_{i+1} \in N(v_i)$: $g[v_{i+1}] \leq g[v_i] + k(v_i, v_{i+1}) \leq g^*(v_i) + k(v_i, v_{i+1}) = g^*(v_{i+1})$ (da der Schlüsselwert $g[v_i]$ während des Laufs des Algorithmus nicht mehr geändert wird).

5. Jede Ecke wird höchstens endlich oft wiederbelebt.

   Bei Wiederbelebung wurde ein kürzerer Weg von S zur Ecke gefunden. Falls G endlich ist, kann dies nur endlich oft geschehen, da es nur endlich viele verschieden lange Wege zu einer Ecke gibt.

   Sei nun G unendlich mit endlichem Außengrad für jede Ecke und $k(r) \geq \varepsilon > 0$ für alle $r \in R$. Sei die Länge des ersten für v gefundenen Weges $p[v]$ gleich K. Jeder günstigere Weg als p besitzt höchstens K/ε Kanten. Da G endlichen Außengrad besitzt, kann es im Falle der Endlichkeit von S nur endlich viele Wege mit Start in S und höchstens K/ε Kanten geben.

6. Der Algorithmus bricht nach endlich vielen Schritten ab. Nach 3 kann der Algorithmus nicht erfolglos abbrechen. □

Satz 1.6 *Seien die Voraussetzungen von Satz 1.5 gegeben und zusätzlich h ein optimistischer Schätzer. Der Algorithmus terminiere mit der Ecke $z \in F$. Dann ist $p[z]$ bei Abbruch ein kostengünstigster Weg von S nach F.*

Beweis: Sei $z \in F$ die Ecke bei erfolgreichem Abbruch. Es gilt:

$$g[z] = g[z] + \underbrace{h(z)}_{=0} = \Phi(z) \leq \Phi(v_m) = g[v_m] + h(v_m)$$

$$\overset{\text{Satz 1.5 Punkt 4}}{=} g^*(v_m) + h(v_m) \leq g^*(v_m) + h^*(v_m) = K^*,$$

Ende der Vorlesung am 18.11

also $g[z] = K^*$, da sowieso $g[z] \geq g^*(z) \geq K^*$ gilt. □

Korollar 1.7 *Unter den Voraussetzungen von Satz 1.6 terminiere der A^*-Algorithmus mit einer Ecke v^* und $K^* := g[v^*]$. Während des Laufs des Algorithmus werden höchstens diejenigen Ecken v mit $\Phi(v) \leq K^*$ expandiert.*

Beweis: Sei v die aktuell expandierte Ecke. Mit den Notationen des Beweise von Satz 1.6 gilt während des Laufs für v_m, daß $g[v_m] = g^*(v_m)$. Somit folgt

$$\Phi(v) \leq \Phi(v_m) = g[v_m] + h(v_m) = g^*[v_m] + h(v_m) \leq g^*(v_m) + h^*(v_m) \leq K^*.$$

□

Lemma 1.8 *Unter den Voraussetzungen von Satz 1.6 terminiere der A^*-Algorithmus mit einer Ecke v^* und $K^* := g[v^*]$. Während des Laufs des Algorithmus wurde jede Ecke v mit $g^*(v) + h(v) < K^*$ expandiert.*

Beweis: Widerspruchsannahme: Für v gelte: $g^*(v) + h(v) < K^*$, aber v werde nicht expandiert.

Wir ändern den Graphen G zu G' durch Hinzufügen einer zusätzlichen Kante (v, t) mit $k(v, t) := h(v)$ ab. Der Schätzer h ist dann auch für G' ein optimistischer Schätzer. Ferner gibt es nun in G' einen Weg p' von S nach T über v der Länge $g^*(v) + h(v) < K^*$.

Wir wenden nun den A^*-Algorithmus auf G' an. Wiederum wird dann v nicht expandiert und der alte, in G optimale, Weg gefunden, obwohl p' ein kürzerer Weg ist. Dies widerspricht der Optimalität des A^*-Algorithmus, die in Satz 1.6 gezeigt wurde. □

Im Worst-Case-Szenario für die Laufzeit des A^*-Algorithmus ist der Schätzer h überall gleich Null, nur nicht auf dem optimalen Weg $(v_0, \ldots, v_r)$, auf dem exakt geschätzt werde: $h(v_i) = h^*(v_i)$, $i = 0, \ldots, r$. Eine Ecke v_i ist dann früh in open, muß aber später aktualisiert werden.

Vorteile durch Schätzer

Während man z.B. den Dijkstra-Algorithmus als Spezialfall mit $h \equiv 0$ erhält, kann man unter bestimmten Voraussetzungen durch geschickte Wahl des Schätzers bessere Laufzeiten erreichen.

Zur Erinnerung: Auf einem Graphen $G = (V, R)$ mit $n := |V|$ Ecken und $m := |R|$ Pfeilen besitzt der Dijkstra-Algorithmus eine Laufzeit von $\mathcal{O}(n \log n + m)$, wenn er mit Fibonacci-Heaps implementiert wird.

Ein *euklidische Graph* ist ein Graph, dessen Eckenmenge Punkten in der euklidischen Ebene entsprechen und bei dem das Gewicht der Kante (v, w) gleich dem euklidischen Abstand $d_2(v, w)$ von v und w ist (ein euklidischer Graph muß nicht vollständig sein). Für diese euklidischen Graphen bietet sich (bei $T = \{t\}$) die optimistische Schätzfunktion $h(v) := d_2(v, t)$ an. Für diese Schätzfunktion konnten Sedgewick und Vitter bei bestimmten Klassen von „zufälligen Graphen“ eine durchschnittliche Laufzeit von $\mathcal{O}(n)$ nachweisen.

Also: Wichtig ist die Wahl einer guten Schätzfunktion.

Im Beispiel des 8-Puzzle sind die in der Vorlesung vorgestellten Distanzfunktionen (d_1, etc.) optimistische Schätzer. Insbesondere findet der A^*-Algorithmus mit diesen optimistischen Schätzern eine kürzeste Zugfolge.

Wahl guter Schätzer

Definition 1.9 (Besser und nicht schlechter informierter Schätzer)
Für zwei optimistische Schätzer h und h' heißt

1. *h' **besser informiert** als h, wenn $h(v) < h'(v) \leq h^*(v)$ für alle $v \in V$ ist.*

2. *h' **nicht schlechter informiert** als h, wenn $h(v) < h'(v) \leq h^*(v)$ für alle $v \in V$ ist.*

Satz 1.10 *Unter den Voraussetzungen von Satz 1.6 seien $A = A^*(h)$ und $A' = A^*(h')$ beides A^*-Algorithmen mit Schätzern h und h', wobei h' besser informiert sei als h. Dann wird jede Ecke, die von A' expandiert wird, auch von A expandiert.*

Beweis: Sei K^* die Länge eines optimalen Weges von S nach F. Es werde die Ecke v von A' expandiert. Nach Korollar 1.7 gilt dann: $\Phi'(v) \leq K^*$. Also haben wir:

$$g^*(v) + h(v) < g^*(v) + h'(v) \leq g'[v] + h'(v) = \Phi'(v) \leq K^*.$$

Nach Lemma 1.8 muß dann v auch von A expandiert werden. □

Definition 1.11 (Monotoner Schätzer)
Sei $G = (V, R)$ und $k \colon R \to \mathbb{R}_{\geq 0}$. Die Funktion $h \colon V \to \mathbb{R}$ heißt monotoner Schätzer, *wenn gilt:*

1. *Für alle $t \in F$ ist $h(t) = 0$.*

2. *Für alle $v \in V$ und alle $w \in N(v)$ gilt:*

$$h(v) \leq k(v, w) + h(w).$$

Durch Induktion ergibt sich aus der letzten Definition das folgende Lemma:

Lemma 1.12 *Sei h ein monotoner Schätzer und p ein Weg von a nach b in G. Dann gilt $h(a) \leq k(p) + h(b)$.*

Satz 1.13 *Seien die Voraussetzungen von Satz 1.5 gegeben und h ein monotoner Schätzer. Dann ist h auch optimistisch.*

Beweis: Sei $v \in V$. Falls $h^*(v) = +\infty$, so ist nichts zu zeigen. Sei daher $h^*(v) < +\infty$ und p ein kürzester Weg von v zu einer Ecke $t \in F$ mit Länge $k(p) = h^*(v)$. Dann gilt:

$$h(v) \overset{\text{Lemma 1.12}}{\leq} k(p) + \underbrace{h(t)}_{=0} = k(p) = h^*(v).$$

□

Bemerkung 1.14 *Ist die Bedingung 1 aus Definition 1.11 nicht erfüllt, so muß ein monotoner Schätzer nicht optimistisch sein: Sei* $F = \{a, b\}$, $S = \{s\}$, $V = S \cup F$, $R = \{(a, b), (s, a)\}$. *Ferner sei* $k(r) = 1$ *für alle* $r \in R$, *sowie* $h(b) = 0$, $h(a) = 1$, $h(s) = 2$. *Dann ist* h *zwar monoton, aber* $h(s) = 2 > 1 = h^*(s)$.

Satz 1.15 *Seien die Voraussetzungen von Satz 1.5 gegeben und* h *ein monotoner Schätzer. Dann gilt:*

1. *Wurde* v' *nach* v *expandiert, dann gilt:* $\Phi(v) \leq \Phi(v')$.
2. *Während des Laufes erfolgt keine Wiederbelebung von Ecken. Es gilt:* $g[v] = g^*(v)$ *für alle* $v \in$ ***closed***.
3. *Wurde* v *expandiert, so galt* $g^*(v) + h(v) \leq K^*$.

Beweis:

1. Wird v expandiert und ist w ein Nachfolger von v, der entweder beim Expandieren neu entdeckt wird oder für den durch das Expandieren von v der Schlüsselwert $g[w]$ verringert wird, dann gilt:

   $$\Phi(w) = g[w] + h(w) = g[v] + k(v, w) + h(w) \overset{h \text{ monoton}}{\geq} g[v] + h(v) = \Phi(v). \tag{1.1}$$

   Wir zeigen nun durch Induktion nach der Anzahl der Expansionen, daß die Φ-Werte der expandierten Ecken monoton wachsen.

   Der Induktionsanfang folgt aus (1.1).

   Im Induktionsschritt sei w die aktuell expandierte Ecke und v die direkt davor expandierte Ecke. Wir unterscheiden zwei Fälle:

   1. Fall: Der Schlüsselwert $g[w]$ ist beim Expandieren von v verringert worden oder w ist beim Expandieren neu entdeckt worden.

   In diesem Fall folgt $\Phi[w] \geq \Phi[v]$ aus (1.1).

   2. Fall: Der Schlüsselwert $g[w]$ wurde nicht verändert.

   Dann galt beim Expandieren von v, daß $\Phi[v] \leq \Phi[w]$, da v die Ecke mit minimalem Φ-Wert war. Diese Ungleichung bleibt weiterhin erhalten, da $g[w]$ nicht geändert wurde.

2. Wenn w wiederbelebt würde, so gäbe es ein v, welches *nach* w als Minimum in Schritt 9 gewählt wird, so daß zum Zeitpunkt der Expansion von v gilt:

   $$g[v] + k(v, w) < g[w].$$

   Nach Punkt 1 gilt $\Phi(v) \geq \Phi(w)$, also $\Phi(w) \leq g[v] + h(v)$. Somit folgt:

   $$\begin{aligned} g[w] &\leq g[w] + \underbrace{h(w) + k(v, w) - h(v)}_{\geq 0, \text{ da } h \text{ monoton}} \\ &= \Phi(w) - h(v) + k(v, w) \\ &\leq g[v] + k(v, w) \\ &< g[w]. \end{aligned}$$

Widerspruch! Somit erfolgt keine Wiederbelebung.

Wir wissen, daß $g[v] \geq g^*(v)$ für alle $v \in V$. Wäre $g[v] > g^*(v)$ für ein $v \in V$, so fände der A^*-Algorithmus keinen günstigsten Weg zu v. Dies kann leicht zum Widerspruch zur der Korrektheit des A^*-Algorithmus geführt werden.

3. Nach Korollar 1.7 gilt für jede expandierte Ecke v, daß $\Phi(v) \leq K^*$. Also folgt

$$K^* \geq \Phi(v) = g[v] + h(v) \geq g^*(v) + h(v).$$

□