Grundlagen der Wissensverarbeitung-Übungsblatt 2

Übungsgruppe 2; Tom Kastek (4kastek@inf), Phil Sehlmeyer (4sehlmey@inf) · WiSe 17/18

1 Exercise 2.3 : (Search Space Construction 2)

Für die Darstellung des Suchraums nutzen wir die euklidische Ebene, die einzelnen Positionen auf dem Brett sind unsere Knoten, jegliche Verkehrsverbindungen (also Taxi, Bus, U-Bahn) werden als Kanten zwischen den Knoten dargestellt.Jeder Punkt erhält zusätzlich Variablen in der Anzahl der Detektive, denen später pro Detektiv eine Einschätzung ihrer Gefährlichkeit (bei der gilt, je geringer der Wert, desto gefährlicher für Mister X, da schneller zu erreichen) zugewiesen wird, zusätzlich noch eine Variable, in der letztendlich die Summe der einzelnen Werte gespeichert wird. Die Devise von Mister X sollte sein, sich möglichst weit von dem Bewegungsbereich der Detektive weg zu bewegen, es gilt also, jeden Knoten eine Einschätzung der Gefahr zuzuteilen, um dann mit der Anzahl A an Schritten eine Position zu erreichen, die eine möglichst geringe Gefahr hat, von den Detektiven erreicht zu werden.

Da die Position der Detektive von Beginn an bekannt ist, können wir einen Markierungsalgorithmus nutzen, um zunächst sämtliche benachbarten Knoten des Startknotens des Detektivs zu finden. Diesen wird anschließend in der dem Detektiv zugewiesenen Variable der Wert 1 gegeben. Der Algorithmus iteriert dann über sämtliche Nachbarknoten der bereits markierten Knoten. Falls ein Knoten gefunden wird, der bereits markiert wurde, wird der Gefahrenwert nicht geändert (er wird ja nicht weniger gefährlich dadurch, dass wir einen längeren Weg zu ihm finden). Auf diesen Weg werden sämtliche Knoten, die von dem Detektiv erreichbar sind, gekennzeichnet.

Dies wird nun für jeden Detektiv durchgeführt. Wenn der Algorithmus für sämtliche Detektive durchgelaufen ist, wird die letzte Variable jedes Knotens mit dem Gesamtwert der Gefahr gefüllt, der Summe der einzelnen Werte.

Die Aufgabe des Suchalgorithmus von Mister X ist es nun, in Abhängigkeit der Schritte A einen erreichbaren Knoten zu finden, der einen möglichst hohen Gesamtwert hat, also möglichst schwer von den Detektiven erreicht werden kann.

2 Exercise 2.3 : (Search Space Construction 3)

Möbel in der Wohnung platzieren

- Search Space: In einem Zustand sind all unsere Möbel gespeichert. Zu den Möbel gehört jeweils eine Platzierung im Raum oder null falls das Möbelstück noch nicht platziert ist. Die Übergänge definieren sich dabei durch das Setzen eines Möeblstücks an einer Position im Raum. Mit jedem Setzen/Ändern einer Position wird diese neu im Zustand zum Möbelstück gespeichert.
- Ziel: Das Ziel ist es, die Möbel so zu platzieren, dass keine der Positionen vor der Tür ist und, dass die Platzierung der Stühle in der Nähe des Tisches ist.
- Properties: Das Ende eines Baums ist, wenn alle Möbel gesetzt sind. Keine Loops, da wir kein Umplatzieren ermöglichen. Wir können also einen direkten Weg suchen.

- bevorzugte Strategie: Heuristische Suche. Mit der heuristischen Suche könnten wir zusätzlich mit einberechnen lassen, dass die Tür nicht platziert werden soll und nach Platzierung der Tische sollen Platzierungen der Stühle als nächstes besser bewertet werden und noch besser die Platzierung in der Nähe des Tisches. (oder nach der Platzierung des ersten Stuhles).

Baustellenplanung

- Search Space: In unserem Zustand sind gespeichert:
  - Teilaufgaben mit der Info, im Bau, fertig, kann nicht gebaut werden. Kann nicht gebaut werden ist die Eigenschaft einer Teilaufgabe, wenn Vorraussetzungen noch nicht erfüllt wurden (wie Wand bemalen ohne, dass eine Wand steht). im Bau ist die Eigenschaft, dass an der Teilaufgabe gebaut werden kann. Also sobald eine Wand steht kann gemalt werden. Fertig ist eine erledigte TEilaufgabe. Wenn also die Wand vollständig bestrichen ist.
  - Auch in unserem Zustand gespeichert sind die einzelnen Arbeiter. Diese können jeweils nur bestimmte Aufgaben erfüllen und werden abhängig davon einer "im Bau" verfügbaren Aufgabe zugeteilt oder auf wartend gesetzt.

  Die Übergänge sind fertigstellungen der Teilaufgaben. Wenn eine Teilaufgabe beendet ist, muss der Zustand in fertig wechseln. Optional wechseln andere Teilaufgaben zu "im Bau" und die Arbeiter müssen sich einer neuen Teilaufgabe zuordnen. Zusätzlich muss den Übergängen eine Wertung in Abhängigkeit von der benötigen Zeit zugeordnet werden.
- Ziel: Ziel ist es mit möglichst wenig Zeitaufwand, also einer kleinen Punktzahl in den Übergängen, das Haus fertig zu stellen.
- Properties: Wir haben keine Loops, da jede Aufgabe nur einmal erfüllt werden kann. Allerdings kann jede Aufgabe auch abhängig von den Arbeitern unterschiedlich schnell erlefigt werden.
- bevorzugte Strategie: Lowest-Cost-First, da es sich hier um ein Problem handelt möglichst geringe Gesamtkosten zu erhalten.

Elevator:

- Search Space: In einem Zustand sind alle erreichbaren Stockwerke des Fahrstuhls gespeichert. Jedes Stockwerk hat einen der Zustände
  - soll erreicht werden
  - soll abgeholt werden
  - kein Interesse
  - aktueller Standort

Außerdem muss jedem Stockwerk ein Wert zugeteilt werden, der die Wartezeit angibt. Stockwerke mit "kein Interesse" oder "aktueller Standord" haben immer den Wert 0. "Soll erreicht werden" und "soll abgeholt werden" haben immer einen Wert größer 0. Als Übergänge wird jedes erreichte Stockwerk genutzt. Mit der Fahrt von A nach B werden also auch alle Stationen dazwischen Übergänge. Nach jedem Übergang steigt jeder Wert über 1 um einen weiteren Punkt. Zu diesem Zeitpunkt wird auch aktualisiert, ob ein Stockwerk den Zustand von kein Interesse zu "Soll abgeholt werden" wechselt. Bei jedem erreichten Stockwerk, wechselt der Zustand des Stockwerks auf kein Interesse. Falls dies ein neuer Zustand ist, kann sich auch der Zustand aller anderen Stockwerke in "Soll erreicht werden" ändern. Immer wenn ein Stockwerk auf "soll erreicht" oder "soll abgeholt werden" wechselt, ändert sich der Wert auf 1. Ein Wechsel von "soll erreicht werden" zu "soll abgeholt werden" oder umgekehrt ist nicht möglich. Falls von einem Stockwerk beides gewünscht wird, soll das keinen Einfluss auf den Wert und damit die weitere Fahrt haben.

- Ziel: Das Ziel ist zu einem den durchschnittlichen Gesamtwert aller Stockwerke möglichst gering zu halten, aber dabei auch kein einzelnen Wert größer X (abhängig von Gesamtstockwerken) werden zu lassen.
- Properties: Dieser Baum ist theoretisch unendlich, da immer wieder neue Ziele erreicht werden sollen. Daher sollte der Baum von einer Berechnung ohne den Wechsel zu "soll erreicht werden" oder "soll abgeholt werden" rechnen.
- bevorzugte Strategie: A* Suche.Diese Suche wurde in der VL bisher nur angeschnitten. Der Beschreibung nach zu urteilen werden Kosten einzelner Übergänge und Gesamtkosten versucht zu berücksichtigen. Unserer einzelnen Pfade sind dabei jeweils die Übergänge von einem Stockwerk bis hin zu einem Stockwerk der erreicht, bzw. abgeholt werden soll. Die Kosten ermitteln sich aus der Steigerung des Gesamtwerts über diesen Pfad. Die Kosten bis zum Ziel werden dabei am besten berechnet, indem man die durchschnittlichen Werte aller Pfade betrachtet. Hier sollte heurisitsch erst die Pfade bevorzugt werden, in denen kein Stockwerk neu zu "soll abgeholt werden" wechselt.