Informed search strategies and local search

Best-first search

De nodes worden geëxpandeerd volgens de waarde van de evaluatie functie.
- Deze evaluatie functie is een kosten schatting.
- Node met de laagste geschatte kost wordt eerst geëxpandeert.

Cfr. UCS

Hangt af van een heuristische functie: h(n).
Een heuristische functie is een duimregel, simplificatie of een beredeneerde gok voor de kost van het goedkoopste pad naar het gewenste doel.

Greedy best-first search

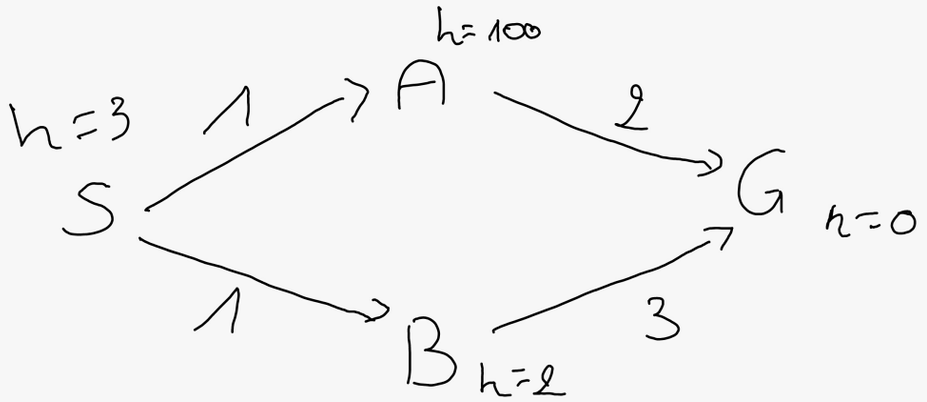
Expandeer altijd de node met de beste heuristiek.

Dit leidt niet noodzakelijk tot de beste oplossing, maar er is wel altijd snel een oplossing.
Ook is iedere geëxpandeerde node deel van de oplossing.
Greedy best-first is ook niet compleet.

A* search

UCS:	Enkel rekening houden met het gewicht
Greedy BFS:	Enkel rekening houden met de heuristiek
A*:	Rekening houden met som van gewicht en heuristiek

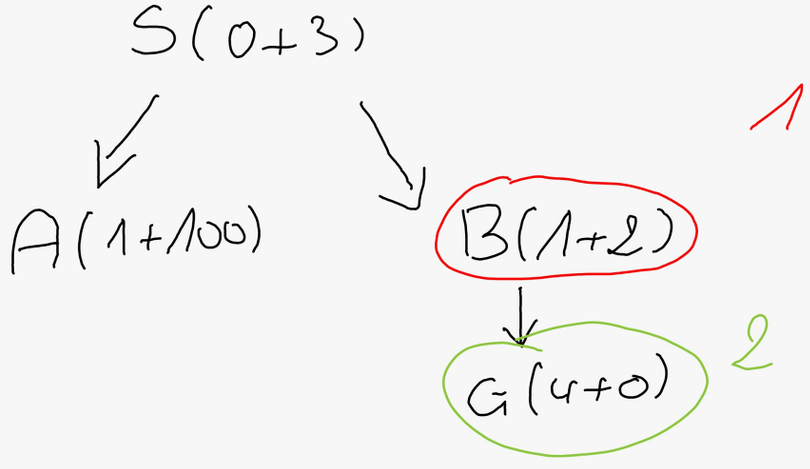
A* evaluatie



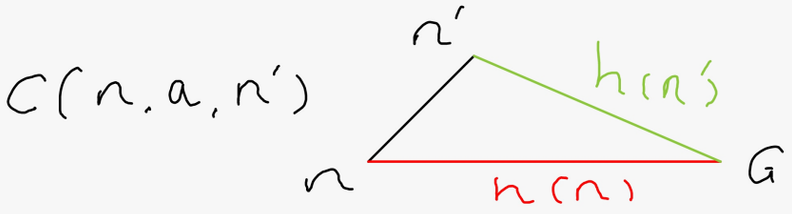
A* evaluatie

Heuristiek in A is zeer pessimistisch:
- h(A) is veel te groot voor de echte waarde
- h(A) is niet toelaatbaar

Een toelaatbare heuristiek overschat de kost nooit.
- Een toelaatbare heuristiek is optimistisch



Consistente heuristiek:
- $h(n) \leqslant c(n,a,n') + h(n')$



Als de heuristiek aanvaarbaar is dan heeft A* de juiste oplossing

A* is compleet
A* is optimaal
A* groeit exponentieel
A* houd alle nodes in het geheugen

Heuristische functies

Heuristische funcites zijn vaak vereenvoudigingen:
- Muren weglaten
- Doen alsof je alleen bent
- Bewegingsrestricties weglaten

8-puzzel
1. aantal misplaatste tegels
2. som van de manhattan afstand van alle misplaatste tegels tot hun juiste plaats

Local seach and optimalisation

Soms is het pad niet belangrijk en enkel de uitkomst. (cfr. 8 queens)

Bij lokaal zoeken wordt enkel 1 staat bijgehouden. Van deze staat gaat men verder naar een volgende staat.

Hill-climbing

Heel beperkt rondkijken.
Je komt altijd in een lokaal maximum terecht
Meestal loop je vast
Andere start locatie gebruiken.

Voorbeeld: 8 queens
1. Zet alle koninginnen random op het bord
2. Bereken voor ieder vakje hoeveel koninginnen elkaar zouden aanvallen als de koningin van die kolom naar dat vakje verplaatst wordt.
3. Kies de kolom met de kleinste waarde erin en verplaats de koningin naar het vakje met de kleinste waarde.

Dit loop meestal snel vast. Een voordeel hiervan is dat het snel vastloop (ca. 2 stappen)