Di 15 Dec. Alex  
  
1e fase Blocking heuristiek is gelukt!

BreadthFirst, pq met heuristiek:  
**Level 4.**   
In 45 stappen, add states: 94.000  
Heuristiek: 2 diep, + 1 per laatste auto in de weg  
In 45 stappen, add states: 86400  
Heuristiek: 2 diep, + 2 per laatste auto in de weg  
**level 3.**  
In 35 stappen, add states: 399  
Heuristiek: 2 diep, + 1 per laatste auto in de weg

Ma 14 Dec. Alex

Depthfirst met heuristiek die 1 geeft aan auto naar rechts en anders 0, die runt een paar minute en checkt 370.000 states. Dan zegt ie alles te hebben gechecked en kapt ie ermee.

Level 4 opgelost met depthfirst met heuristiek dat naar rechts: depth + 30, anders: depth  
stoppen op 1e oplossing. Hij deed er 130.000 added states over, 600k gen states en length: 1001  
  
level 4 dept, heuristiek zelfde als hierboven maar nu een straf van 1 als de rode achteruit gaat. Nu in 80000 added states en 300k generated. Length was weer 1001  
  
level 4 met breadth en p.q. in 45 stappen en 130.000 add states en 1.1 mil gen.   
level 5 met breadth en p.q. memory error en

Level 5 met depthfirst zonder heuristiek maar wel de heuristiek file opgelost in 8540 stappen! Met 60 000 stappen.   
level 4 met breadth en p.q. in 45 stappen en 116.000 added states en 1.00 mil gen.  
Dit was na toevoeging van de aantal auto’s in directe lijn met de uitgang heuristiek. Het aantal auto’s werd opgeteld bij de score  
  
Hier hebben we een artikel die wat zegt over rushhour heuristieken  
<http://aaai.org/ocs/index.php/SOCS/SOCS10/paper/viewFile/2079/2522>

Ma 16 Nov. Alex

Ik heb BreadthFirst search geimplementeerd. Op de 2e 6 bij 6 test case. Dit levert 1 oplossing op, met lengte 26. Volgens profiler kost dit 0.62 seconden Dit is met python zijn ingebouwde Queue.Queue class. Deze class handelt ook threading, dat ie het stukje geheugen blocked enzo  
Wanneer deze queue vervangen word door een simpele list geeft dit een totaal van: 0.58 seconden

Zo 15 Nov. Alex

Ik heb bounded DFS geimplementeerd hij stelt nu elke keer als ie een oplossing vind zijn maximale diepte bij zodat ie alleen maar betere oplossingen vind.   
Ik heb de childcount methode geimplementeerd om bij te houden op welke level men zit in de possibility tree. Dit houd ook bij welke opeen volgende stappen zijn genomen om te komen waar men op dat moment is. Dit gebeurd met hulp van de PathTracker class.