Rush-hour

PSPACE, ANSI, State

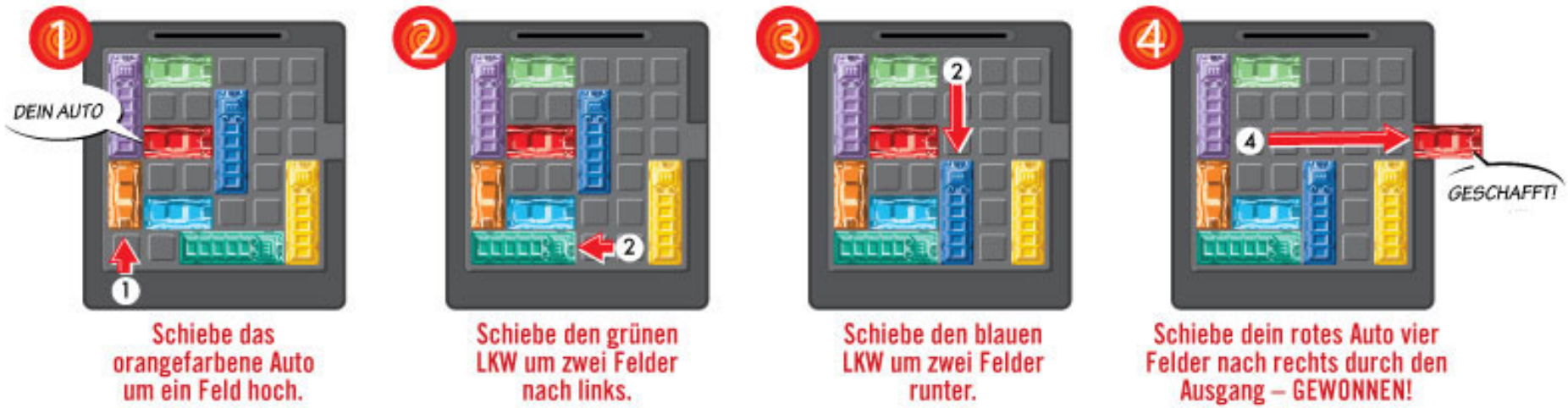
Christian Höner zu Siederdissen
christian.hoener.zu.siederdissen@uni-jena.de

Theoretische Bioinformatik, Bioinformatik Uni Jena

November, 2023

Rush-hour

https://www.thinkfun.de/products/rush-hour/



Rushhour ist *PSPACE-complete* und damit wahrscheinlich nur in exponentieller Zeit lösbar

Generische Puzzle-Spiele

Es handelt sich hierbei um Spiele für die Folgendes gilt:

- Spiele laufen in diskreten Zügen ab
- zu jedem beliebigen Zeitpunkt gibt es wohldefinierte Spielzustände
- aufeinander folgende Zustände lassen sich jeweils exakt beschreiben

Rush-hour ist so ein Spiel, aber auch Sudoko, Schach, go, Mensch ärgere dich nicht, und viele mehr. Wir interessieren uns nur für "1-Spieler" Spiele, da dann nicht "reagiert" werden muss.

Generische Puzzle-Spiele

Gegeben den aktuellen Zustand S eines Spiels, sind drei Probleme zu lösen (teilweise mittels Moves M):

solved $S \to \mathbb{B}$ is *wahr* wenn S ein Lösungszustand ist

move $S \to M \to S$, gegebn S und M wird ein neuer Zustand S generiert

moves $S \to \{M\}$, gegeben S, generiere alle legalen Moves $M_1 \ldots M_S$ für S

Allerdings hängen die Typen von S und M vom jeweiligen Spiel ab: wie organisieren wir das?

Spielkoordinaten

```
 1     1   2   3   4   5   6
 2
 3     8   9  10  11  12  13
 4
 5    15  16  17  18  19  20 ==> -- Ausgang
 6
 7    22  23  24  25  26  27
 8
 9    29  30  31  32  33  34
10
11    36  37  38  39  40  41
```

Definiere und löse Rush-hour

```haskell
-- | Noetig fuer die Typ-klasse
data Rushhour = Rushhour

-- | Zellen sind einfach 'Int's
type Cell = Int

-- | Grid ist eine Liste von Zell-paaren
-- fuer erste, letzte Zelle
type Grid = [(Cell,Cell)]

-- | Auto-Index
newtype VehicleIx = VIx Int
  deriving (Eq,Ord,Show,Enum,Num)
```

BFS: Breitensuche

- erweitert in jedem "moves" alle noch aktiven Pfade
- deren Anzahl exponentiell wächst
- insbesondere wächst die Anzahl der Intermediär-Zustände exponentiell, da *alle* Pfade der Länge $n-1$ besucht werden, bevor der Lösungspfad der Länge n gegangen
- dafür ist aber n minimal

DFS: Tiefensuche

- erweitert in jedem "moves" alle noch aktiven Pfade
- deren Anzahl auch exponentiell wächst
- es wächst auch die Anzahl der Intermediärzustände exponentiell, aber wahrscheinlich langsamer
- insbesondere wird der "lexikographisch" (was immer das hier bedeutet) erste Pfad gefunden, der löst
- die Pfadlänge n kann aber sehr lang werden

Zahlen, bitte!

- Es sind drei Probleme g_0, g_1, g_2 gegeben
- |Pfad| ist die Länge der Lösung
- |States| wie viele Zwischen-States besucht wurden
- sec. die Zeit um die Lsg. zu finden

	BFS			DFS		
	|Pfad|	|States|	sec.	|Pfad|	|States|	sec.
g_0	8	646	0.10	67	66	0.00
g_1	35	2757	0.55	1229	2094	0.31
g_2	82	3051	0.61	1306	1675	0.17

Aufgaben

`https://www.michaelfogleman.com/rush/`

- Download der Datenbank
- Parsen der Datenbank
- Konvertierung in das Format hier *oder* eigene Rush-hour Definition
- Problem und Daten verstehen
- Diese Version kennt auch "Mauern" die nicht beweglich sind!

Wer sich wirklich austoben möchte: generiert eigene Probleme für "beliebige" $N \times N$ große Instanzen.