Algorithmen in Zellularautomaten

7. Synchronisation

Thomas Worsch

Fakultät für Informatik
Karlsruher Institut für Technologie

Sommersemester 2017

Überblick

- Das klassische Firing Squad Synchronization Problem
- General an beliebiger Stelle
- Zweidimensionales FSSP
- Ausblick

Problem (Firing Squad Synchronisation Problem, FSSP)

- **Gegeben:** $Q' = \{\#, \mathtt{g}, \mathtt{s}, \mathtt{f}\}$ und $N = H_1^{(1)}$
- **Gesucht:** Zellularautomat mit $Q \supseteq Q'$ und δ, so dass gilt:
  - $\{\#, \mathtt{s}\}$ ist passiv und # ist tot.
  - C überführt jede Konfiguration der Form $\#\mathtt{gss}\cdots\mathtt{s}\#$ in die Konfiguration $\#\mathtt{fff}\cdots\mathtt{f}\#$ mit gleichem Träger,
  - und zwar so, dass dabei in keiner der vorher auftretenden Konfigurationen der Zustand f vorkommt.

- **Beachte:**
  - Aktivität kann sich nur vom „General" g ausbreiten
  - immer gleiches Q, gleiches δ unabhängig von der Größe des Trägers

Problem (Firing Squad Synchronisation Problem, FSSP)

- **Gegeben:** $Q' = \{\texttt{\#}, \texttt{g}, \texttt{s}, \texttt{f}\}$ und $N = H_1^{(1)}$
- **Gesucht:** Zellularautomat mit $Q \supseteq Q'$ und δ, so dass gilt:
  - $\{\texttt{\#}, \texttt{s}\}$ ist passiv und # ist tot.
  - C überführt jede Konfiguration der Form $\texttt{\#gss} \cdots \texttt{s\#}$ in die Konfiguration $\texttt{\#fff} \cdots \texttt{f\#}$ mit gleichem Träger,
  - und zwar so, dass dabei in keiner der vorher auftretenden Konfigurationen der Zustand $\texttt{f}$ vorkommt.

- **Beachte:**
  - Aktivität kann sich nur vom „General" $\texttt{g}$ ausbreiten
  - immer gleiches Q, gleiches δ unabhängig von der Größe des Trägers

Geschichte

- laut Waksman (1966):
  „The problem known as the "firing squad synchronization problem" was devised about the year 1957 by J. Myhill. It first appeared in print in a paper by E. F. Moore […]"
  „The problem first arose in connection with causing all parts of a self-reproducing machine to be turned on simultaneously, and was first solved by J. McCarthy and M. Minsky."
- erste „zitierfähige" Lösung:
  Eiichi Goto:
  A minimal time solution of the firing squad problem.
  Dittoed course notes for Applied Mathematics 298,
  Harvard University, (1962), pp. 52-59.

Algorithmus (Balzer)

Algorithmus (Balzer)

(1> >									
(2>	>								
(3>		>							
(	1>		>						
(	2>			>					
(	3>				>				
(		1>				>			
(		2>					>		
(		3>						>	
(			1>						<)
(			2>					<	)
(			3>				<		)
(				1>		<			)
(				2>	<				)
(				< <1)	(1> >				)

Algorithmus (Balzer)

(1> >									
(2>	>								
(3>		>							
(	1>		>						
(	2>			>					
(	3>				>				
(		1>				>			
(		2>					>		
(		3>						>	
(			1>						<)
(			2>					<	)
(			3>				<		)
(				1>		<			)
(				2>	<				)
(				< <1)	(1> >				)
(			<	<2)	(2>	>			)
(		<		<3)	(3>		>		)
(	<		<1	)	(	1>		>	)
(>			<2	)	(	2>			<)
(	>		<3	)	(	3>		<	)
(		«1)(1»		)	(		«1)(1»		)

Algorithmus (Balzer)

(1> >									
(2>	>								
(3>		>							
(	1>		>						
(	2>			>					
(	3>				>				
(		1>				>			
(		2>					>		
(		3>						>	
(			1>						<)
(			2>					<	)
(			3>				<		)
(				1>		<			)
(				2>	<				)
(				< <1)	(1> >				)
(			<	<2)	(2>	>			)
(		<		<3)	(3>		>		)
(	<		<1	)	(	1>		>	)
(>			<2	)	(	2>			<)
(	>		<3	)	(	3>		<	)
(		«1)(1»		)	(		«1)(1»		)
(	<	<2)(2>	>	)	(	<	<2)(2>	>	)
(>		<3)(3>		<)	(>		<3)(3>		<)
(	«1)(1»	)(	«1)(1»	)	(	«1)(1»	)(	«1)(1»	)

Algorithmus (Balzer)

(1> >	s	s	s	s	s	s	s	s	s
(2>	>	s	s	s	s	s	s	s	s
(3>		>	s	s	s	s	s	s	s
(	1>		>	s	s	s	s	s	s
(	2>			>	s	s	s	s	s
(	3>				>	s	s	s	s
(		1>				>	s	s	s
(		2>					>	s	s
(		3>						>	s
(			1>						<)
(			2>					<	)
(			3>				<		)
(				1>		<			)
(				2>	<				)
(				< <1)	(1> >				)
(			<	<2)	(2>	>			)
(		<		<3)	(3>		>		)
(	<		<1	)	(	1>		>	)
(>			<2	)	(	2>			<)
(	>		<3	)	(	3>		<	)
(		«1)(1»		)	(		«1)(1»		)
(	<	<2)(2>	>	)	(	<	<2)(2>	>	)
(>		<3)(3>		<)	(>		<3)(3>		<)
(	«1)(1»	)(	«1)(1»	)	(	«1)(1»	)(	«1)(1»	)
f	f	f	f	f	f	f	f	f	f

Zeitbedarf (von Balzers Algorithmus)

für die Synchronisation von n Zellen „ungefähr“

$$t(n) \approx \frac{3}{2}n + t(\frac{n}{2}) \ .$$

Also

$$t(n) = 3n + \text{Terme niedrigerer Ordnung} \ .$$

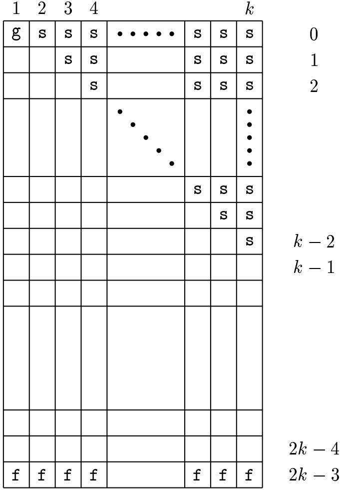
Optimal?

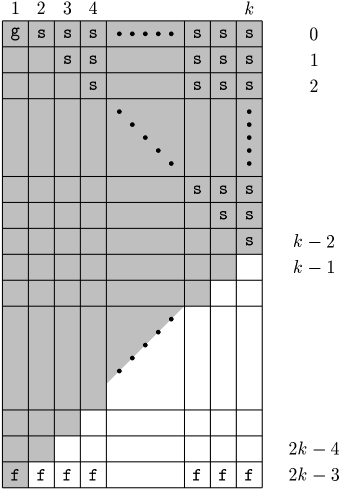
Satz (Waksman, Goto, Mazoyer)

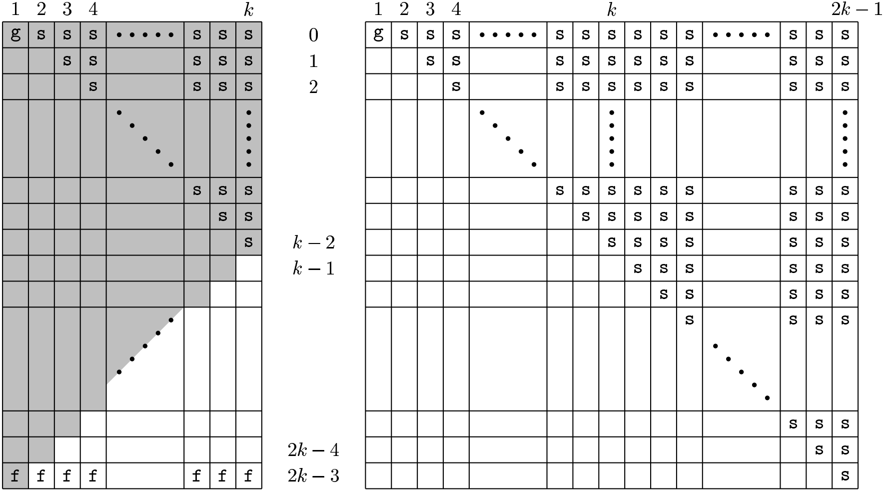
1. Es gibt keinen Zellularautomaten, der das FSSP für alle n löst und für ein $k \geq 2$ höchstens $2k - 3$ Schritte benötigt (Waksmann, 1966).
2. Es gibt einen Zellularautomaten, der das FSSP für alle $n \geq 2$ in genau $2n - 2$ Schritten löst (Goto, 1962).
   Man kommt dabei mit nur 7 Zuständen aus (Mazoyer, 1987) also außer `#`, g, `s`, `f` nur 3 weitere.

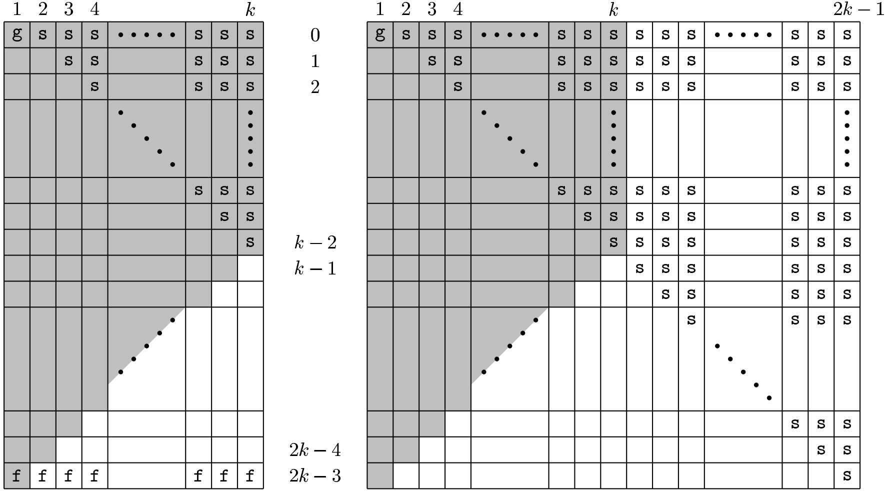
Beweis

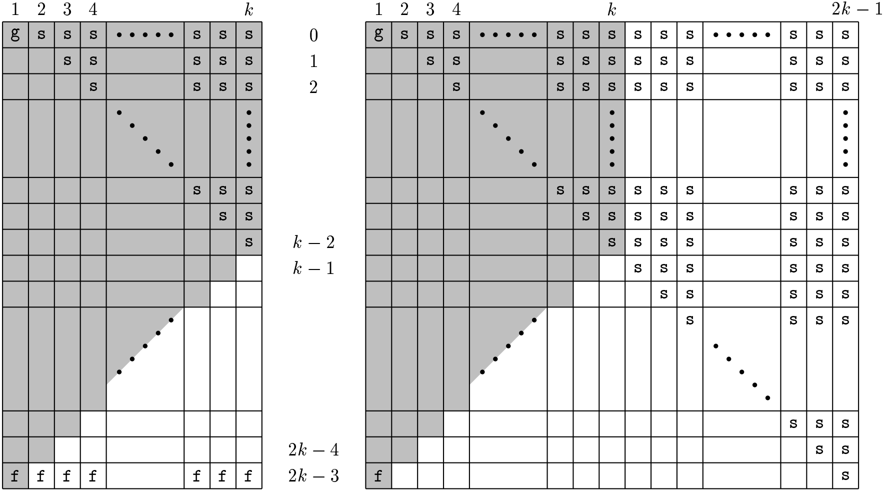
- obere Schranke: Algorithmen; siehe später
- untere Schranke: indirekt
  - **Gegeben:** ein Zellularautomat C
  - **Annahme:** C löst das FSSP für ein k in höchstens $2k-3$ Schritten
  - **Zeige:** C löst das FSSP nicht für alle n

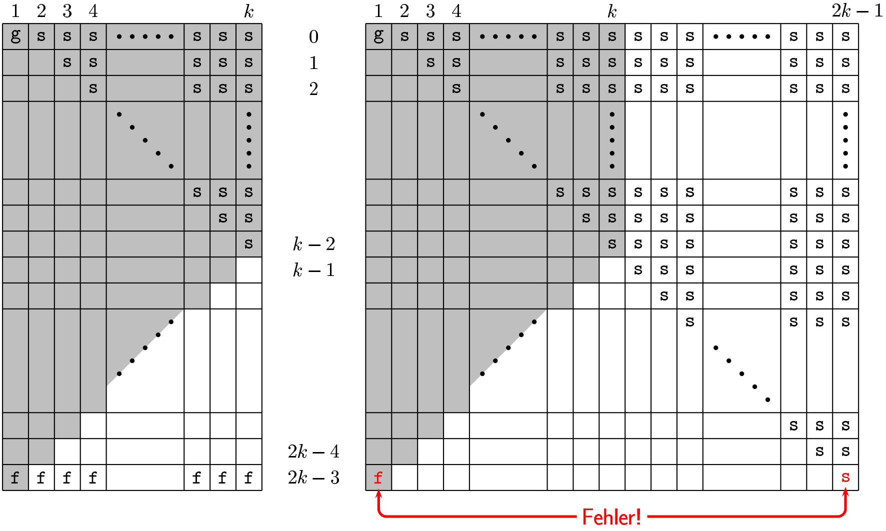

1 2 3 4 k
g s s s • • • • • s s s
s s s s s
s s s s
s s s
s s
s
0
1
2
k − 2
k − 1
2k − 4
2k − 3
f f f f f f f


1 2 3 4 k
g s s s • • • • • s s s
0
1
2
k − 2
k − 1
2k − 4
2k − 3
f f f f f f f

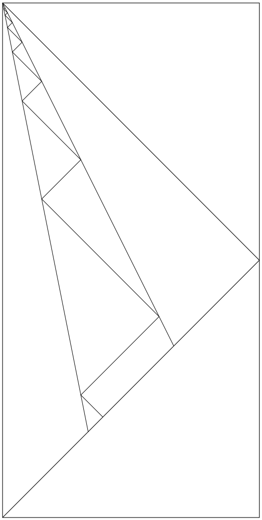

1 2 3 4 k
g s s s s s s
s s s s s
f f f f f f f
0
1
2
k − 2
k − 1
2k − 4
2k − 3
1 2 3 4 k 2k − 1


1 2 3 4 k
1 2 3 4 k 2k − 1
g
s
f
0
1
2
k − 2
k − 1
2k − 4
2k − 3

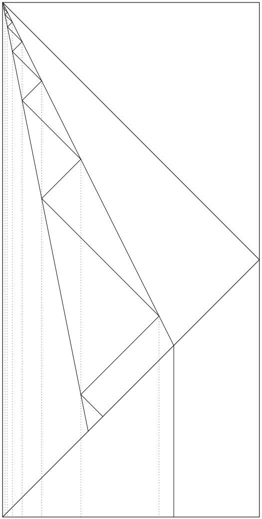

1 2 3 4 k
0
1
2
k − 2
k − 1
2k − 4
2k − 3
1 2 3 4 k 2k − 1


1 2 3 4 k
g s s s • • • • • s s s
0
1
2
k − 2
k − 1
2k − 4
2k − 3
f f f f f f f
1 2 3 4 k 2k − 1
g s s s • • • • • s s s s s s • • • • • s s s
f
s
Fehler!

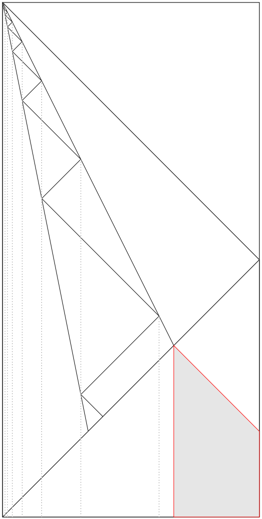
Algorithmus (Gerken)



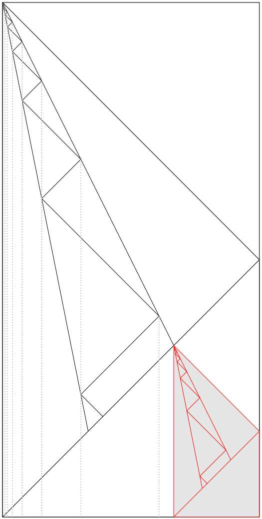
Algorithmus (Gerken)



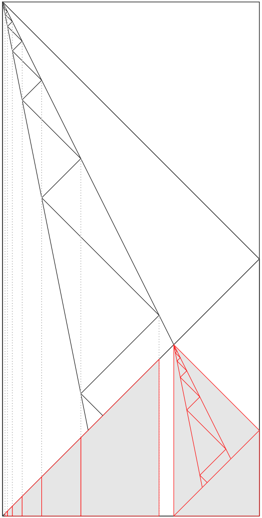
Algorithmus (Gerken)



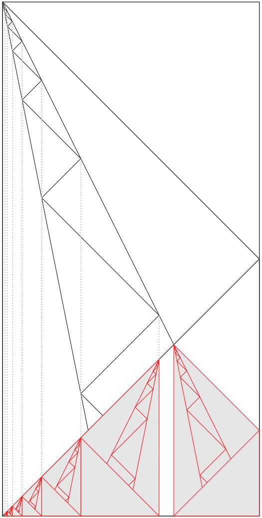
Algorithmus (Gerken)



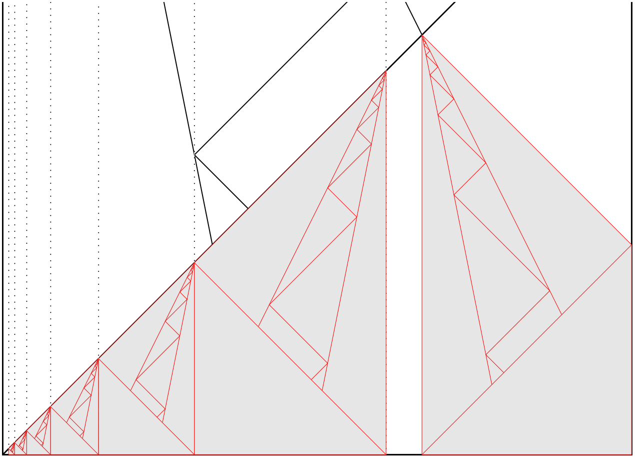
Algorithmus (Gerken)



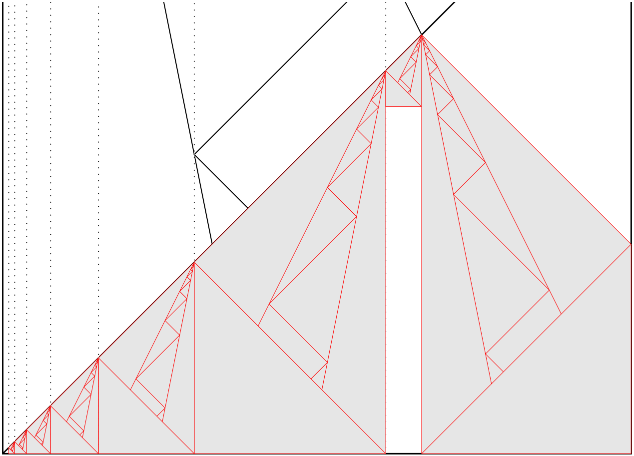
Algorithmus (Gerken)



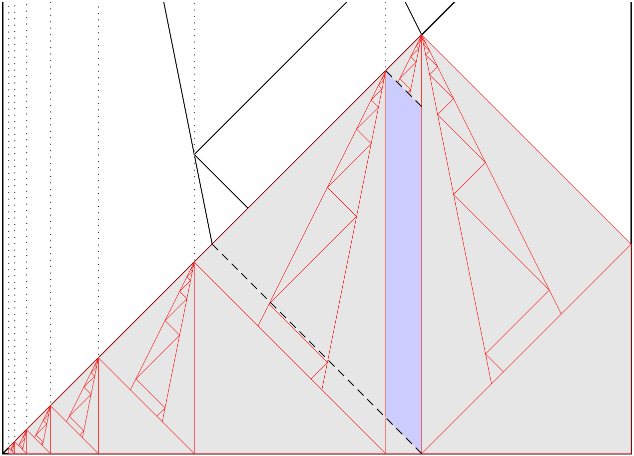
Algorithmus (Gerken)



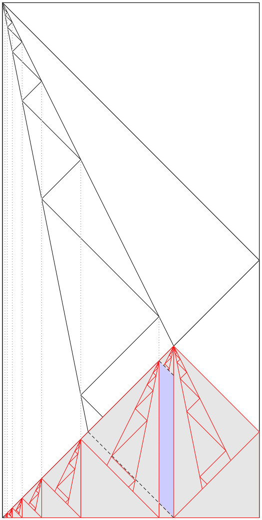
Algorithmus (Gerken)



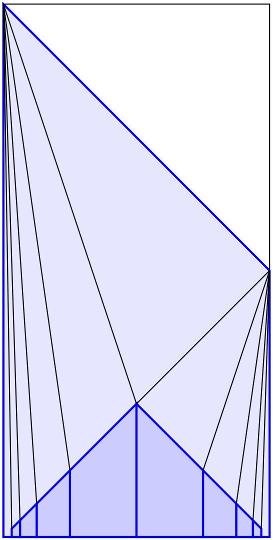
Algorithmus (Gerken)



Algorithmus (Gerken)



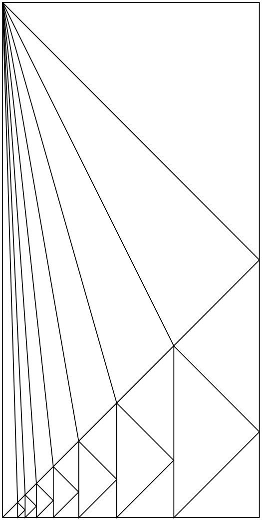
Algorithmus (Waksman)



Algorithmus (Viele verschieden schnelle Signale)

>											
	<>										
1		<>									
1	<0		<>								
1	0	<		<>							
1	1		<		<>						
1	1	<0		<		<>					
1	<0	0	<		<		<>				
1	0	1		<		<		<>			
1	0	1	<0		<		<		<>		
1	0	<	0	<		<		<		<>	
1	1		1		<		<		<		<>
1	1		1	<0		<		<		<	
1	1		<	0	<		<		<		<
1	1	<0		1		<		<		<	
1	<0	0		1	<0		<		<		<
1	0	0		<	0	<		<		<	
1	0	0	<		1		<		<		<
1	0	1			1	<0		<		<	
1	0	1			<	0	<		<		<
1	0	1		<		1		<		<	
1	0	1	<0			1	<0		<		<
1	0	<	0			<	0	<		<	
1	1		0		<		1		<		<
1	1		0	<			1	<0		<	
1	1		1				<	0	<		<
1	1		1			<		1		<	
1	1		1		<			1	<0		<
1	1		1	<0				<	0	<	
1	1		<	0			<		1		<
1	1	<0		0		<			1	<0	
1	<0	0		0	<				<	0	<
1	0	0		1				<		1	
1	0	0		1			<			1	<0
1	0	0		1		<				<	0
1	0	0		1	<0				<		1

Algorithmus (Mazoyer)



Algorithmus (Mazoyer)

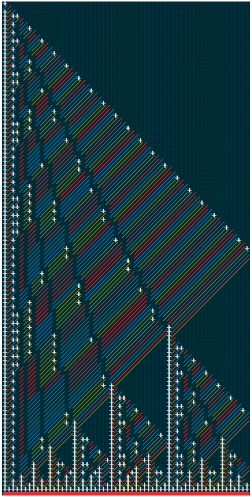


Bild: C. Wellenbrock (60 Zellen)

und nun … ? …

und nun … ? …

Verallgemeinerungen

und nun … ? …

Verallgemeinerungen

Welche fallen Ihnen ein?

Überblick

- Das klassische Firing Squad Synchronization Problem
- General an beliebiger Stelle
- Zweidimensionales FSSP
- Ausblick

Problem (FSSP, General an beliebiger Stelle)

- **Gegeben:** $Q' = \{\texttt{\#}, \mathrm{g}, \texttt{s}, \texttt{f}\}$ und $N = H_1^{(1)}$
- **Gesucht:** Zellularautomat mit $Q \supseteq Q'$ und δ, so dass gilt:
  - $\{\texttt{\#}, \texttt{s}\}$ ist passiv und # ist tot.
  - C überführt
    - jede Konfiguration der Form $\texttt{\#ss} \cdots \texttt{sgs} \cdots \texttt{s\#}$
    - unabhängig von der Position von g
    - in die Konfiguration $\texttt{\#fff} \cdots \texttt{f\#}$
    - mit gleichem Träger,
  - und zwar so, dass dabei in keiner der vorher auftretenden Konfigurationen der Zustand $\texttt{f}$ vorkommt.

Problem (FSSP, General an beliebiger Stelle)

- **Gegeben:** $Q' = \{\#, \mathrm{g}, \mathtt{s}, \mathtt{f}\}$ und $N = H_1^{(1)}$
- **Gesucht:** Zellularautomat mit $Q \supseteq Q'$ und δ, so dass gilt:
  - $\{\#, \mathtt{s}\}$ ist passiv und # ist tot.
  - C überführt
    - jede Konfiguration der Form $\mathtt{\#ss} \cdots \mathtt{sgs} \cdots \mathtt{s\#}$
    - unabhängig von der Position von g
    - in die Konfiguration $\mathtt{\#fff} \cdots \mathtt{f\#}$
    - mit gleichem Träger,
  - und zwar so, dass dabei in keiner der vorher auftretenden Konfigurationen der Zustand $\mathtt{f}$ vorkommt.

Wie kann man das machen?

Satz

Es sei n die Größe des Trägers und k die Länge des kürzeren Abschnittes neben g.

- Das FSSP mit dem General an beliebiger Stelle ist in $2n - 2 - k$ Schritten lösbar.
- Diese Zeit ist für $n \geq 2$ optimal.

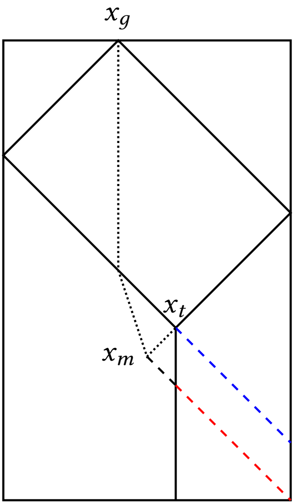
Beweis

- obere Schranke: Algorithmen; siehe gleich
- untere Schranke: analog zu vorhin

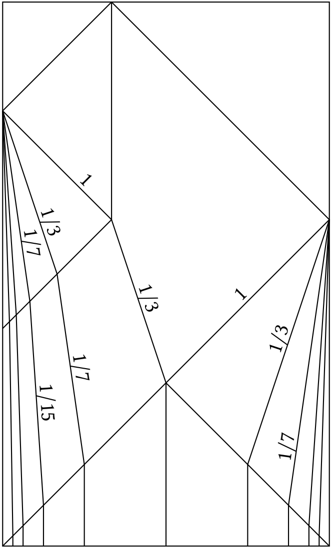
Jedes Ende muss „wissen“, wie weit weg das andere ist.

!! Nicht nur der General muss Bescheid wissen,
deswegen reicht auch *nicht* Zeit $2(n-k)-2$.

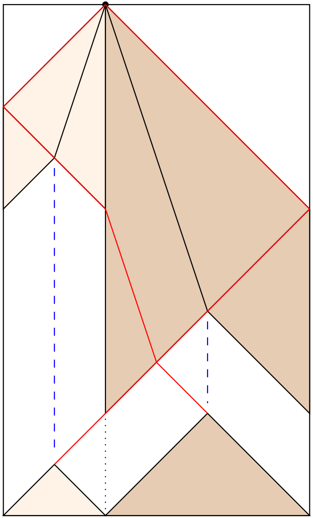
Algorithmus (TW)



Algorithmus (H. Umeo)



Algorithmus (S. Wacker)



Überblick

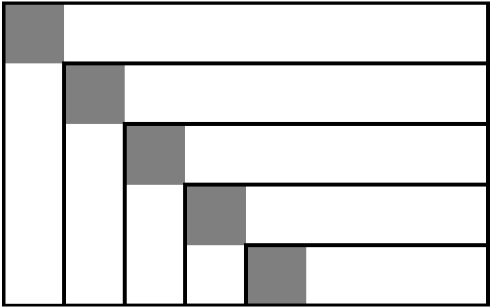
- Das klassische Firing Squad Synchronization Problem
- General an beliebiger Stelle
- Zweidimensionales FSSP
- Ausblick

Problem (FSSP für Rechtecke)

- analog zum Eindimensionalen
  - von Neumann Nachbarschaft (Radius 1)
- Anfangskonfiguration:
  Rechteck von s-Zellen mit
  einer g-Zelle im z.B. linken oberen Eck

g	s	s	s	s	s	s	s
s	s	s	s	s	s	s	s
s	s	s	s	s	s	s	s
s	s	s	s	s	s	s	s
s	s	s	s	s	s	s	s

Algorithmus



Zeitbedarf: $m + n + \max\{m, n\} - 3$ Schritte

Überblick

- Das klassische Firing Squad Synchronization Problem
- General an beliebiger Stelle
- Zweidimensionales FSSP
- Ausblick

Weitere Fragestellungen

- Verallgemeinerungen
  - d-dimensionale Quader mit dem General an beliebiger Stelle (Szwerinski, 1982)
  - mehrere Generäle (Diplomarbeit H. Schmid, 2003)
  - beliebige d-dimensionale zusammenhängende Muster
  - (langsam) wachsende Muster
- Spezialisierungen
  - Quadrate bzw. Würfel
- Modifikationen
  - Early-Bird-Problem (Rosenstiehl, Katona/Legendi, Vollmar)

Zusammenfassung

- Beim eindimensionalen FSSP ist für jeden der Fälle
  - ein General am linken Ende oder an beliebiger Stelle
  - mehrere Generäle an beliebigen Stellen

  zeitoptimale Lösungen bekannt.
- Im zweidimensionalen Fall ist das nur für die Fälle eines Generals bekannt.