

Algorithmen II

Abschlussveranstaltung am 13.02.2014

Zusammenfassung & Anmerkungen zur Klausur

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER



Klausur

Organisatorisches

Was muss ich tun um an der Hauptklausur teilzunehmen?

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)

Normalfall: Anmeldung über das Studierendenportal

Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.

Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)

Normalfall: Anmeldung über das Studierendenportal

Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.

Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).

- Nicht mehr abmelden. (Eine Abmeldung ist noch unmittelbar vor Beginn der Klausur möglich.)

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)

Normalfall: Anmeldung über das Studierendenportal

Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.

Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).

- Nicht mehr abmelden. (Eine Abmeldung ist noch unmittelbar vor Beginn der Klausur möglich.)
- Ab 20. Februar auf der Vorlesungshomepage nachschauen, welchem Hörsaal ihr zugeteilt seid.

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)

Normalfall: Anmeldung über das Studierendenportal

Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.

Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).

- Nicht mehr abmelden. (Eine Abmeldung ist noch unmittelbar vor Beginn der Klausur möglich.)
- Ab 20. Februar auf der Vorlesungshomepage nachschauen, welchem Hörsaal ihr zugeteilt seid.
- Am 24. Februar um 13:45 Uhr bei dem entsprechenden Hörsaal erscheinen. Um 14:00 Uhr startet die 2-Stündige Klausur.

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)
Normalfall: Anmeldung über das Studierendenportal
Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.
Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).
- Nicht mehr abmelden. (Eine Abmeldung ist noch unmittelbar vor Beginn der Klausur möglich.)
- Ab 20. Februar auf der Vorlesungshomepage nachschauen, welchem Hörsaal ihr zugeteilt seid.
- Am 24. Februar um 13:45 Uhr bei dem entsprechenden Hörsaal erscheinen. Um 14:00 Uhr startet die 2-Stündige Klausur.

lernen nicht vergessen!

Was muss ich tun um an der Hauptklausur teilzunehmen?

- Bis zum **17. Februar** anmelden. (Spätere Anmeldung nicht möglich!)
Normalfall: Anmeldung über das Studierendenportal
Informationswirte u. ä.: Anmeldung im Studienbüro: ihr werdet entweder direkt eingetragen oder bekommt einen blauen Schein, den ihr in unserem Sekretariat abgeben müsst.
Schülerstudenten: Anmeldung mit einem formlosen Schreiben (unterschrieben im Sekretariat abgeben).
- Nicht mehr abmelden. (Eine Abmeldung ist noch unmittelbar vor Beginn der Klausur möglich.)
- Ab 20. Februar auf der Vorlesungshomepage nachschauen, welchem Hörsaal ihr zugeteilt seid.
- Am 24. Februar um 13:45 Uhr bei dem entsprechenden Hörsaal erscheinen. Um 14:00 Uhr startet die 2-Stündige Klausur.

lernen nicht vergessen!

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.
- **Nicht** die Themen, die nur in der Übung behandelt wurden. Genauer:
 - Ein Problem, das in der Übung, nicht aber in der Vorlesung behandelt wurde, wird als unbekannt angenommen.
 - Das heißt nicht, dass das Problem nicht in der Klausur definiert werden kann um in der Vorlesung gelernte Techniken darauf zu übertragen.

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.
- **Nicht** die Themen, die nur in der Übung behandelt wurden. Genauer:
 - Ein Problem, das in der Übung, nicht aber in der Vorlesung behandelt wurde, wird als unbekannt angenommen.
 - Das heißt nicht, dass das Problem nicht in der Klausur definiert werden kann um in der Vorlesung gelernte Techniken darauf zu übertragen.

Womit kann ich lernen?

(außer mit diesen Folien hier)

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.
- **Nicht** die Themen, die nur in der Übung behandelt wurden. Genauer:
 - Ein Problem, das in der Übung, nicht aber in der Vorlesung behandelt wurde, wird als unbekannt angenommen.
 - Das heißt nicht, dass das Problem nicht in der Klausur definiert werden kann um in der Vorlesung gelernte Techniken darauf zu übertragen.

Womit kann ich lernen?

(außer mit diesen Folien hier)

- Vorlesungsfolien & Literaturhinweise (auf der Homepage)

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.
- **Nicht** die Themen, die nur in der Übung behandelt wurden. Genauer:
 - Ein Problem, das in der Übung, nicht aber in der Vorlesung behandelt wurde, wird als unbekannt angenommen.
 - Das heißt nicht, dass das Problem nicht in der Klausur definiert werden kann um in der Vorlesung gelernte Techniken darauf zu übertragen.

Womit kann ich lernen?

(außer mit diesen Folien hier)

- Vorlesungsfolien & Literaturhinweise (auf der Homepage)
- Übungsblätter & Übungsfolien (auf der Homepage)

Was erwartet mich in der Klausur?

- Themen, die in der Vorlesung behandelt wurden.
 - Definierte Probleme.
 - Vorgestellte Algorithmen.
 - Anwendung der gelernten Techniken auf andere Problemstellungen.
- **Nicht** die Themen, die nur in der Übung behandelt wurden. Genauer:
 - Ein Problem, das in der Übung, nicht aber in der Vorlesung behandelt wurde, wird als unbekannt angenommen.
 - Das heißt nicht, dass das Problem nicht in der Klausur definiert werden kann um in der Vorlesung gelernte Techniken darauf zu übertragen.

Womit kann ich lernen?

(außer mit diesen Folien hier)

- Vorlesungsfolien & Literaturhinweise (auf der Homepage)
- Übungsblätter & Übungsfolien (auf der Homepage)
- Alte Algo II / Algorithmentechnik Klausuren (auf der Homepage)

Achtung: Die Algorithmentechnik Klausuren waren 1-stündig und der Stoff stimmt nicht komplett überein!

- **Hinweise** in der Klausur:
 - ▶ Aufgaben können auch ohne Hinweise gelöst werden, aber Hinweise liefern hilfreiche Informationen.
 - ▶ Hinweise werden nicht ohne Grund gegeben.

- Teilaufgaben nicht notwendigerweise nach Schwierigkeit sortiert.
 - ▶ Wenn man bei einer Teilaufgabe nicht weiterkommt, nicht die folgenden missachten.

- Lösungswege helfen bei der Korrektur die Aufgabe differenziert zu bewerten.
 - ▶ Bei falscher Antwort: Lösungswege können Teilpunkte geben, die man ohne Lösungsweg nicht bekommt. (Ausgenommen sind Aufgaben zum Ankreuzen.)

- Antworten bei Ankreuzaufgaben gut überlegen.
 - ▶ gibt es Gegenbeispiele, welche die gegebene Antwort widerlegt.
 - ▶ kann man die Antwort beweisen/gut begründen.

Änderung der Regelung:

Es dürfen eigene Wörterbücher verwendet werden (Sprache beliebig).

- Die Wörterbücher dürfen **keine** Änderungen zum original Druck aufweisen.
z.B. handschriftliche Notizen, eingeklebte Ausdrücke.

Die Wörterbücher müssen spätestens bis zum 21. Februar bei uns abgegeben werden.

- Zettel mit Name und Matrikelnummer einlegen!

Die Aufgaben der Klausur sind in Deutsch gestellt.

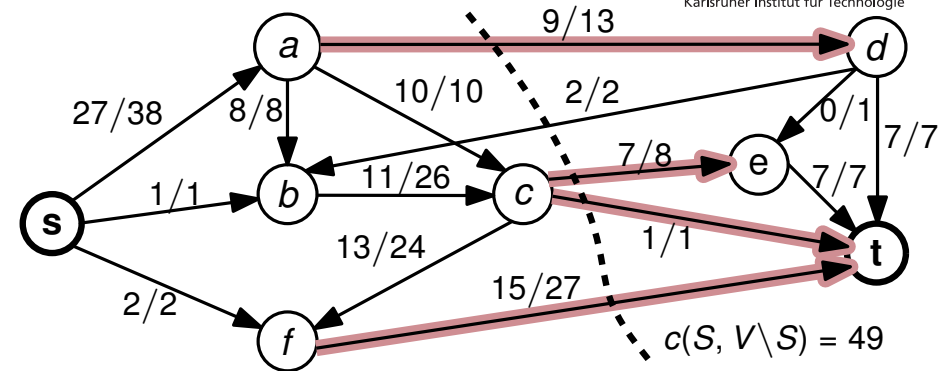
Zusammenfassung

(keine Gewährleistung für inhaltliche Vollständigkeit)

Flüsse & Schnitte

Grundlegendes:

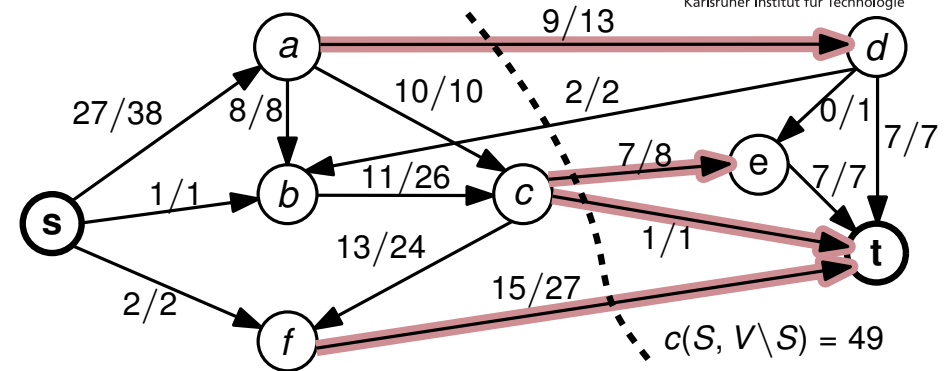
- Was ist ein Flussnetzwerk?
- Welche Bedingungen stellt man an einen Fluss?
- Was ist ein minimaler Schnitt? Und was hat das mit Flüssen zu tun?
- Was ist das Residualnetzwerk und was hat es mit erhöhenden Wegen auf sich?



Flüsse & Schnitte

Grundlegendes:

- Was ist ein Flussnetzwerk?
- Welche Bedingungen stellt man an einen Fluss?
- Was ist ein minimaler Schnitt? Und was hat das mit Flüssen zu tun?
- Was ist das Residualnetzwerk und was hat es mit erhöhenden Wegen auf sich?



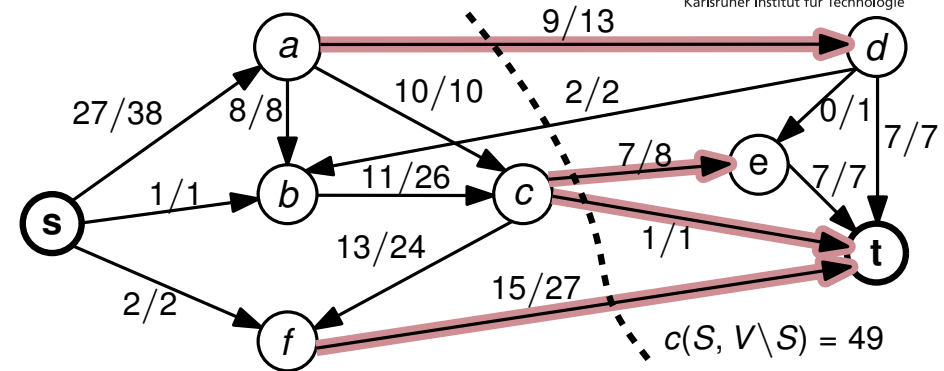
Algorithmen & Techniken:

- Ford-Fulkerson bzw. Edmonds-Karp:
 - Wie funktioniert's?
 - Was sind die Unterschiede?
 - Wie wirkt sich das auf die Laufzeit aus?

Flüsse & Schnitte

Grundlegendes:

- Was ist ein Flussnetzwerk?
- Welche Bedingungen stellt man an einen Fluss?
- Was ist ein minimaler Schnitt? Und was hat das mit Flüssen zu tun?
- Was ist das Residualnetzwerk und was hat es mit erhöhenden Wegen auf sich?



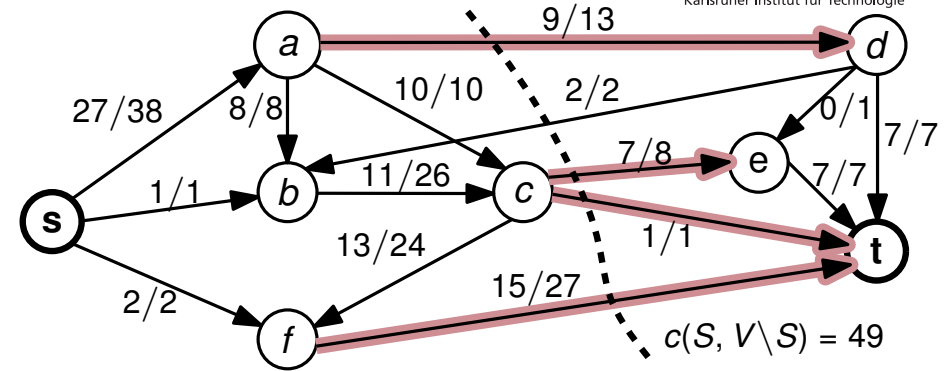
Algorithmen & Techniken:

- Ford-Fulkerson bzw. Edmonds-Karp:
 - Wie funktioniert's?
 - Was sind die Unterschiede?
 - Wie wirkt sich das auf die Laufzeit aus?
- Was ist ein lineares Programm (LP)? Wie kann man Flussprobleme mittels LP lösen?

Flüsse & Schnitte

Grundlegendes:

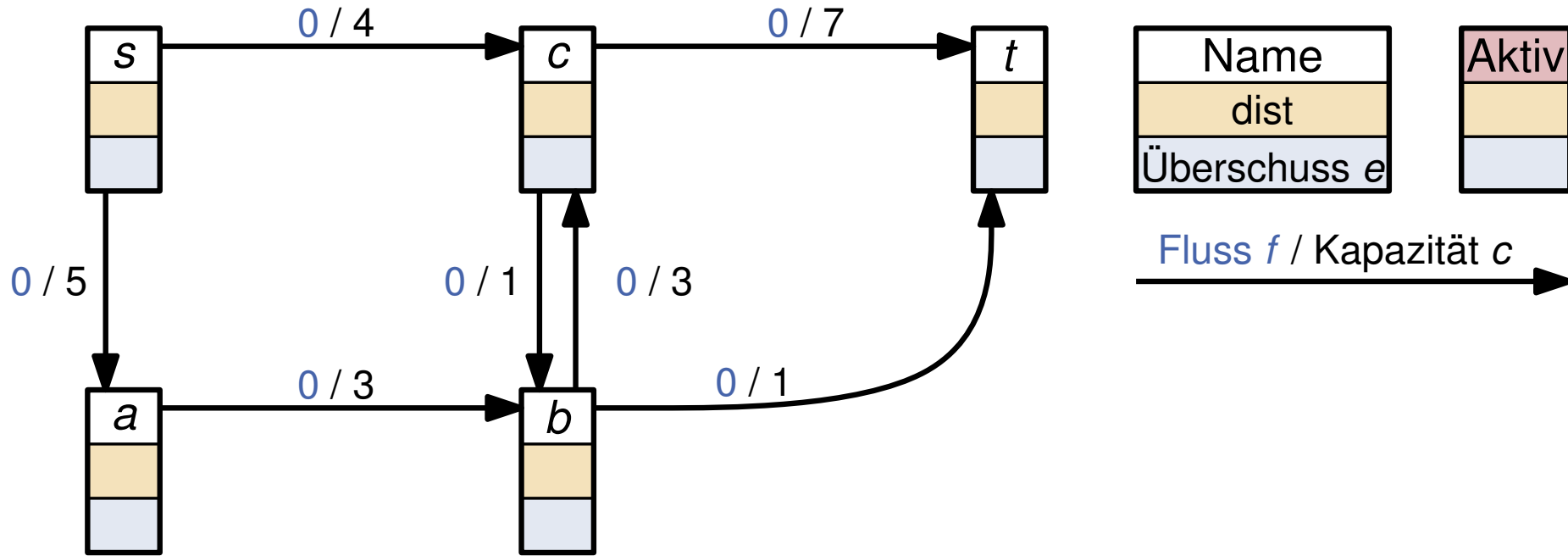
- Was ist ein Flussnetzwerk?
- Welche Bedingungen stellt man an einen Fluss?
- Was ist ein minimaler Schnitt? Und was hat das mit Flüssen zu tun?
- Was ist das Residualnetzwerk und was hat es mit erhöhenden Wegen auf sich?



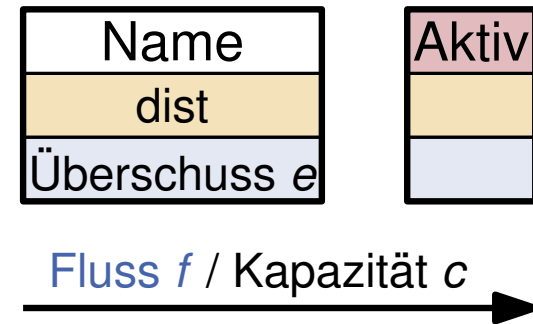
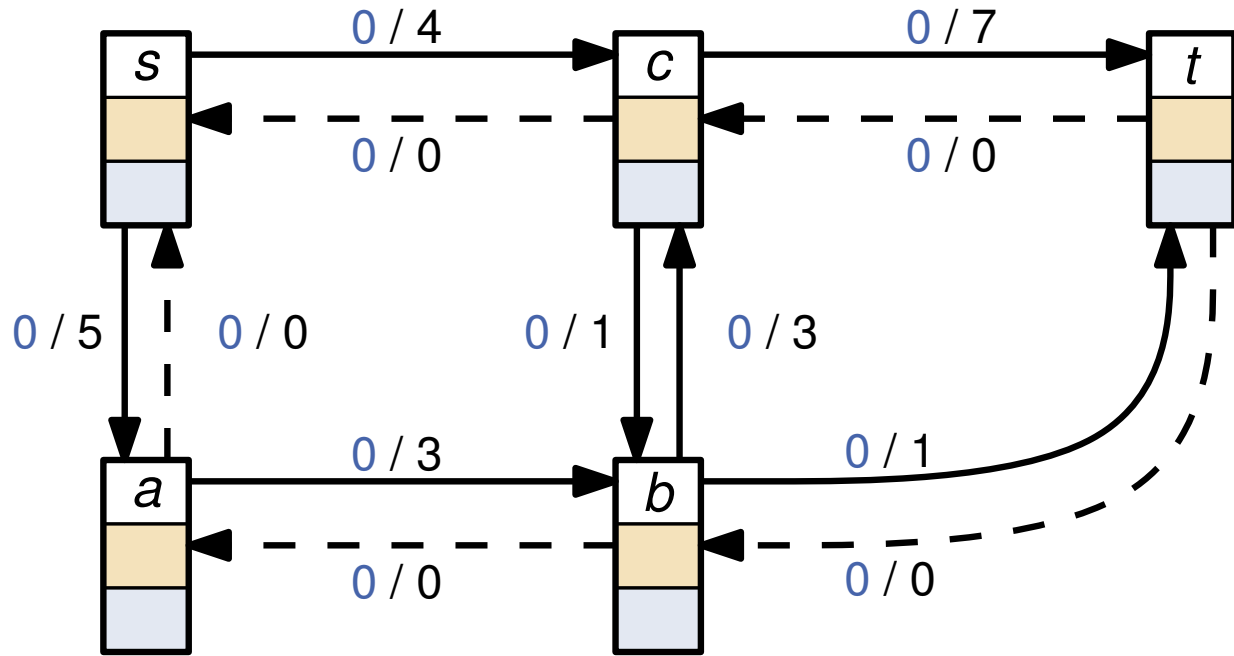
Algorithmen & Techniken:

- Ford-Fulkerson bzw. Edmonds-Karp:
 - Wie funktioniert's?
 - Was sind die Unterschiede?
 - Wie wirkt sich das auf die Laufzeit aus?
- Was ist ein lineares Programm (LP)? Wie kann man Flussprobleme mittels LP lösen?
- Algorithmus von Goldberg & Tarjan: siehe nächste Folie!

Goldberg & Tarjan

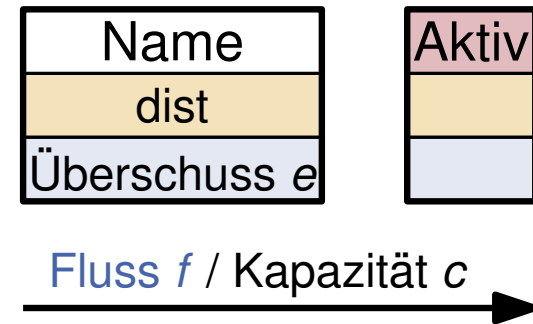
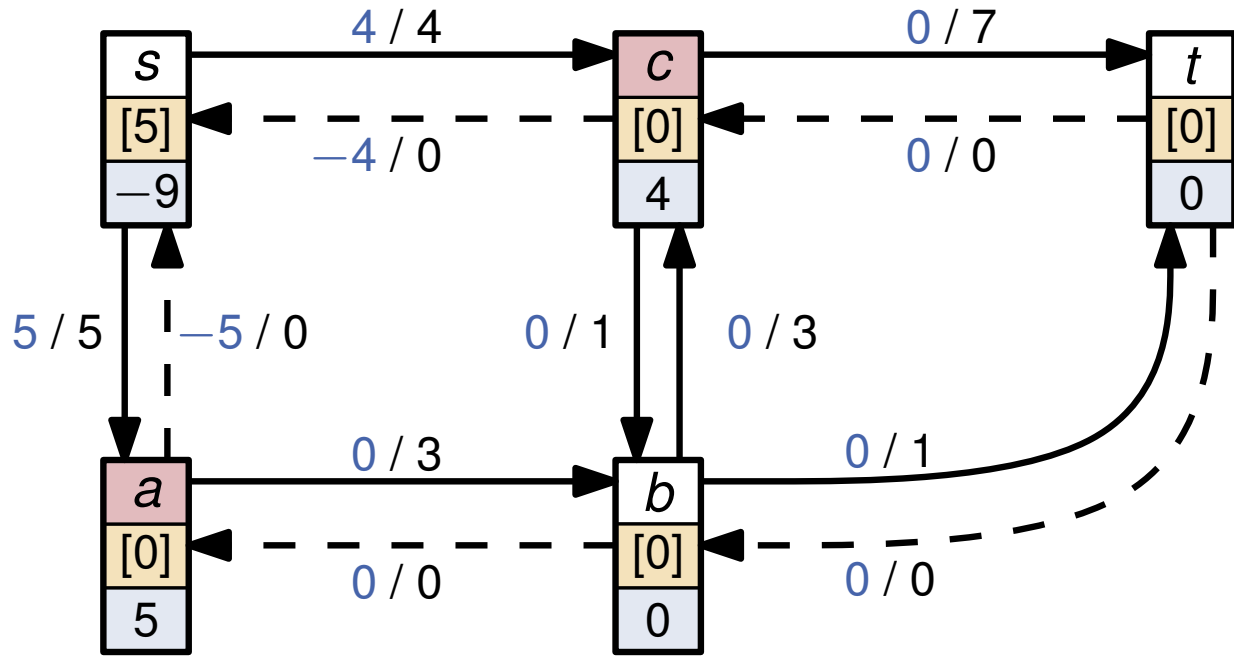


Goldberg & Tarjan



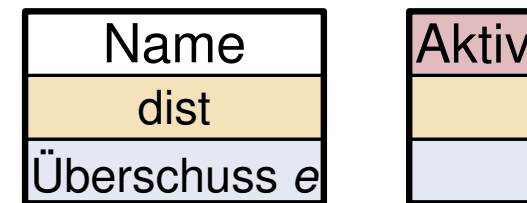
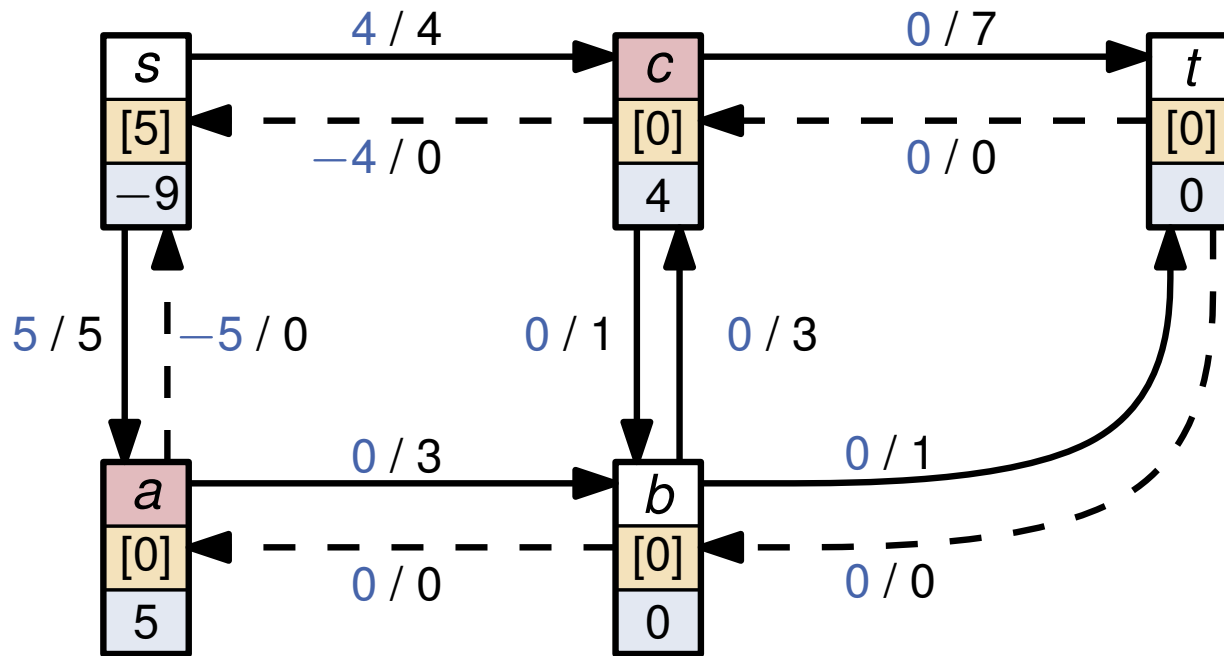
- Führe Gegenkanten ein

Goldberg & Tarjan



- Führe Gegenkanten ein
- Initialisiere dist und *f*

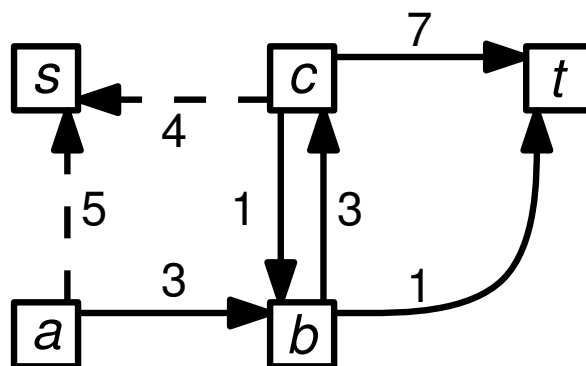
Goldberg & Tarjan

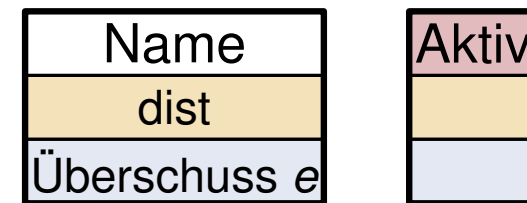
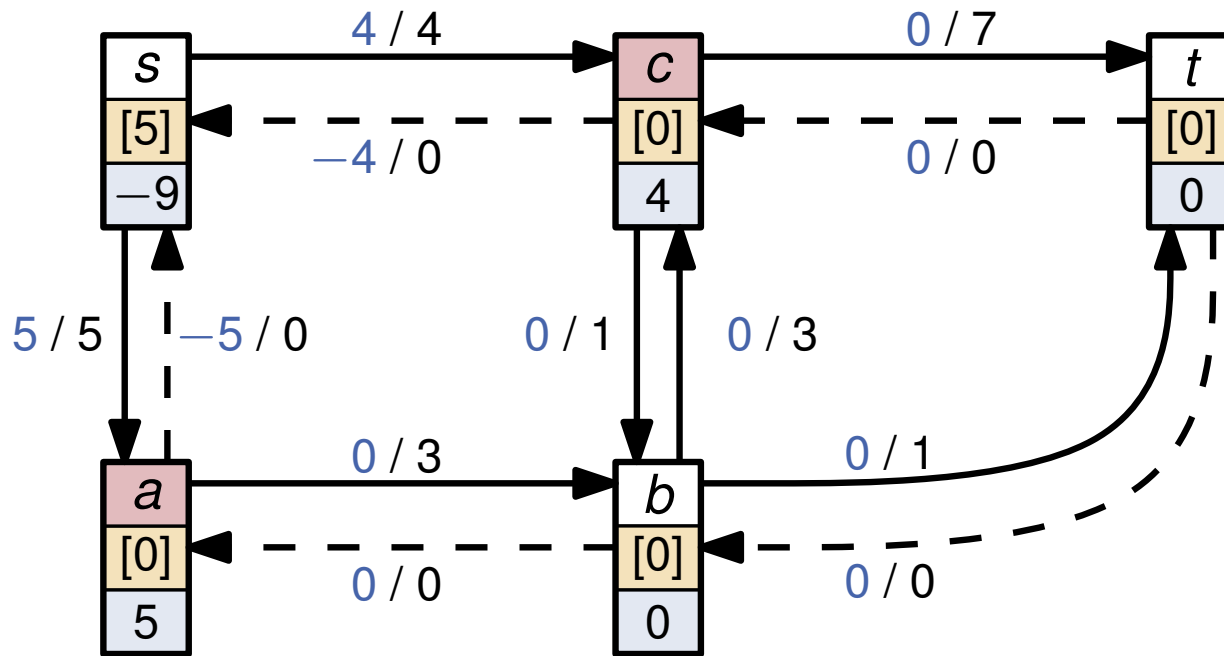


Fluss f / Kapazität c

- Führe Gegenkanten ein
- Initialisiere dist und f
- Betrachte Residualnetzwerk

Residualnetzwerk D_f :

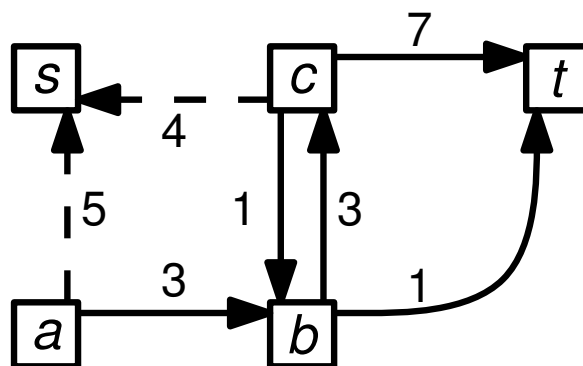


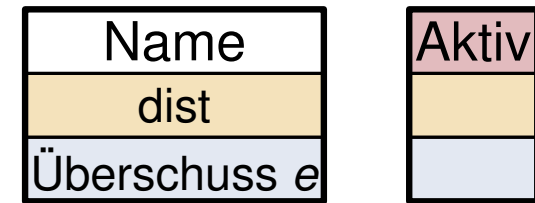
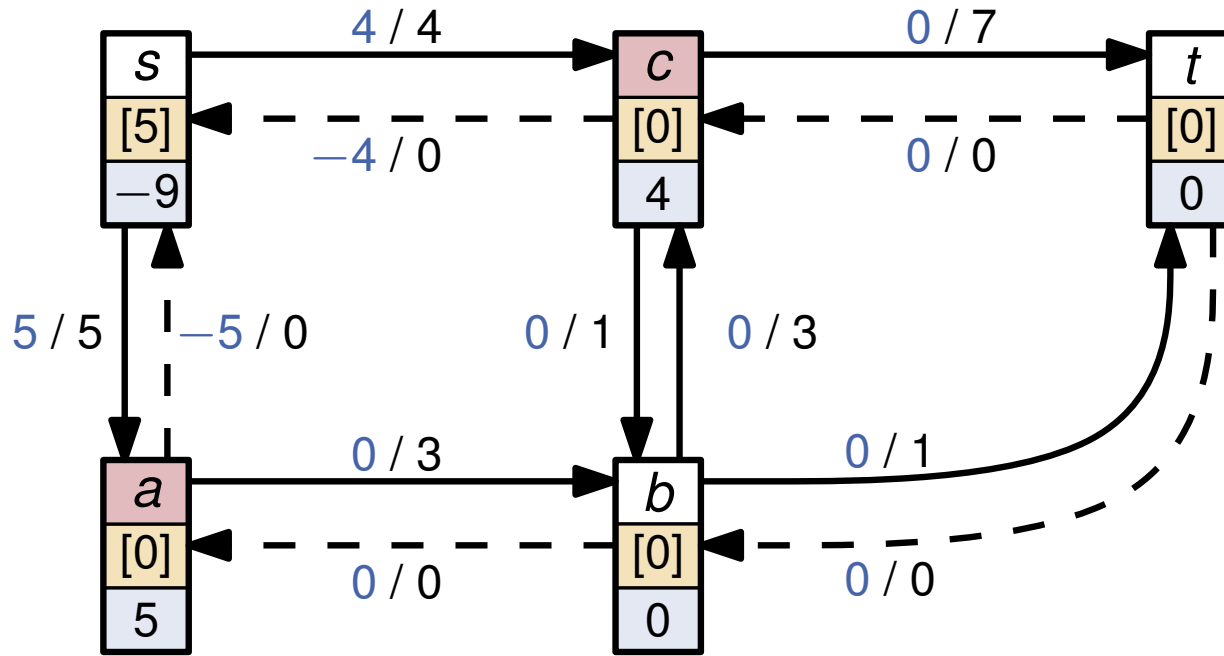


- Führe Gegenkanten ein
- Initialisiere dist und f
- Betrachte Residualnetzwerk

Wähle aktiven Knoten und führe PUSH / RELABEL aus:

Residualnetzwerk D_f :

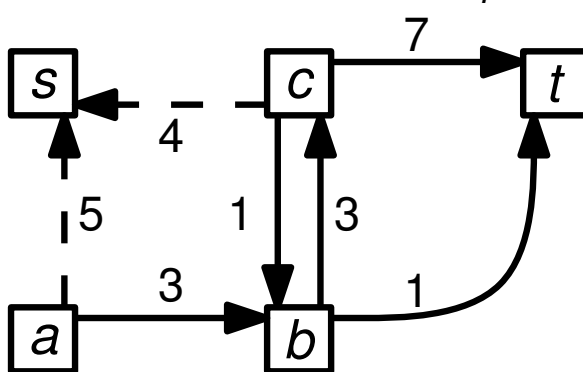




Fluss f / Kapazität c

- Führe Gegenkanten ein
- Initialisiere dist und f
- Betrachte Residualnetzwerk

Residualnetzwerk D_f :



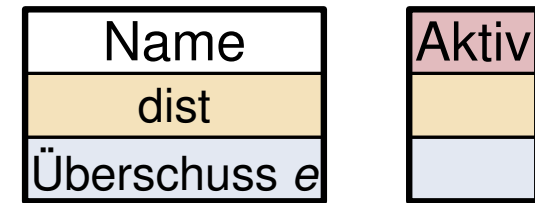
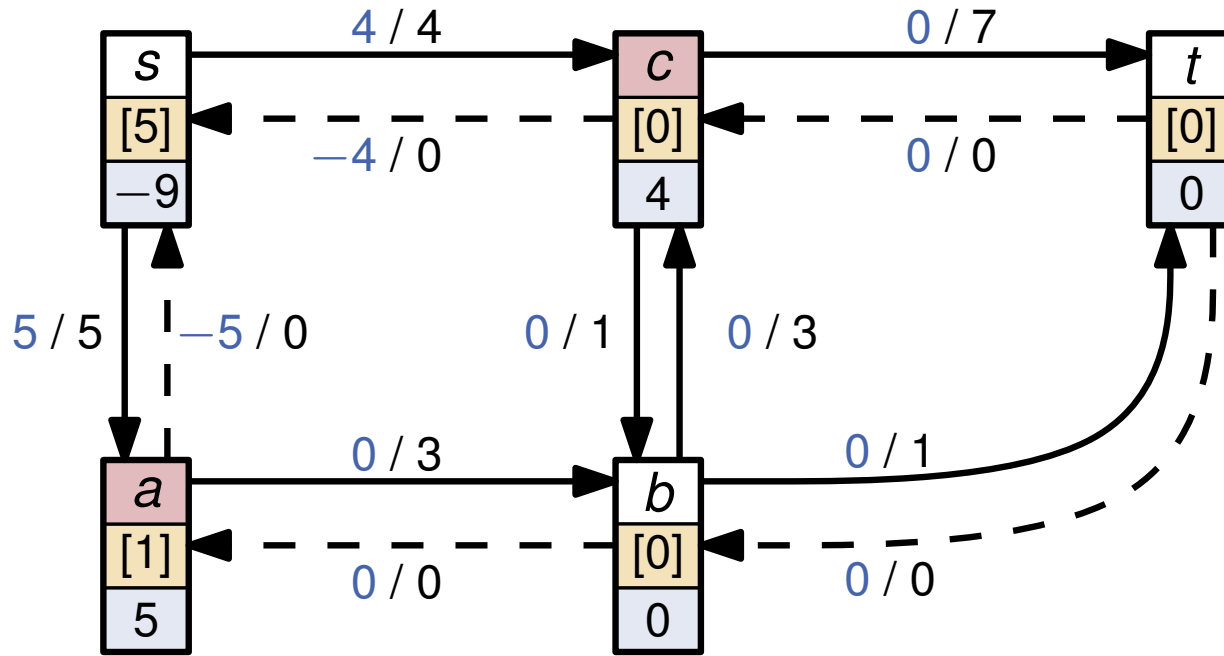
Wähle aktiven Knoten und führe PUSH / RELABEL aus:

Bedingungen zum Ausführen von RELABEL für a :

- Für alle Kanten (a, v) in D_f gilt: $\text{dist}(a) \leq \text{dist}(v)$

Setze Label $\text{dist}(a)$ auf...

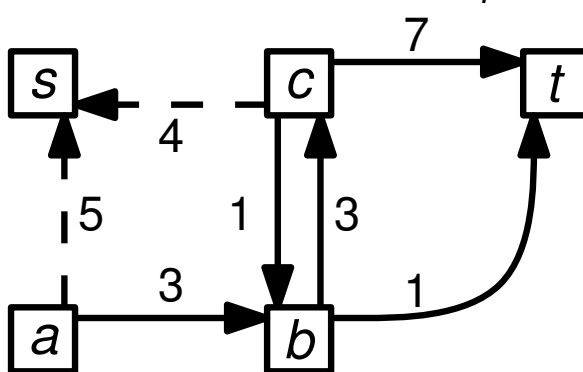
- ∞ , falls es in D_f keine Kante (a, v) gibt.
- Minimum von $\text{dist}(v) + 1$ über alle Kanten (a, v) in D_f .



Fluss f / Kapazität c

- Führe Gegenkanten ein
- Initialisiere $dist$ und f
- Betrachte Residualnetzwerk

Residualnetzwerk D_f :



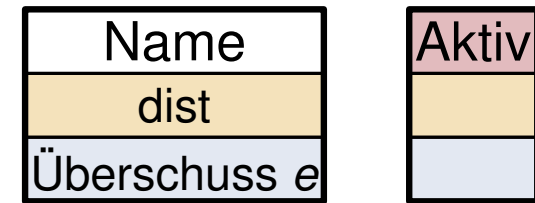
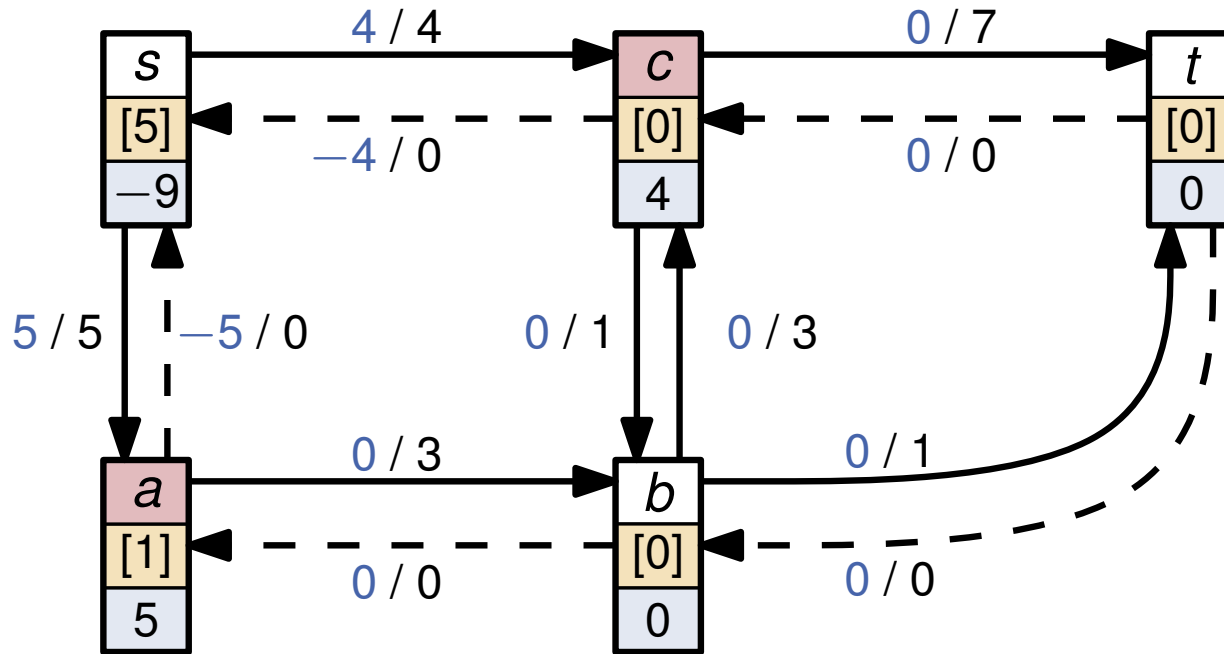
Wähle aktiven Knoten und führe PUSH / RELABEL aus:

Bedingungen zum Ausführen von RELABEL für a :

- Für alle Kanten (a, v) in D_f gilt: $dist(a) \leq dist(v)$

Setze Label $dist(a)$ auf...

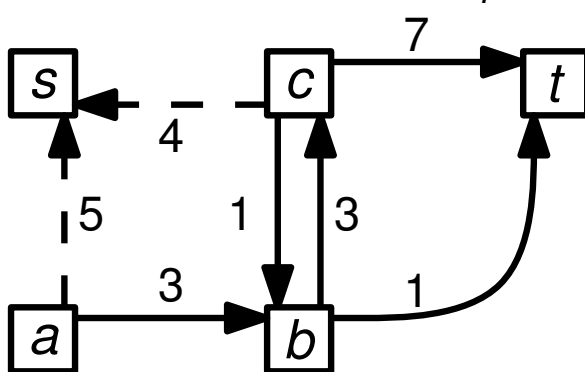
- ∞ , falls es in D_f keine Kante (a, v) gibt.
- Minimum von $dist(v) + 1$ über alle Kanten (a, v) in D_f .



Fluss f / Kapazität c

- Führe Gegenkanten ein
- Initialisiere dist und f
- Betrachte Residualnetzwerk

Residualnetzwerk D_f :

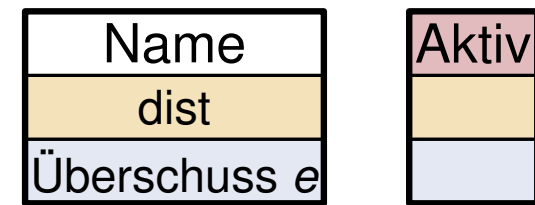
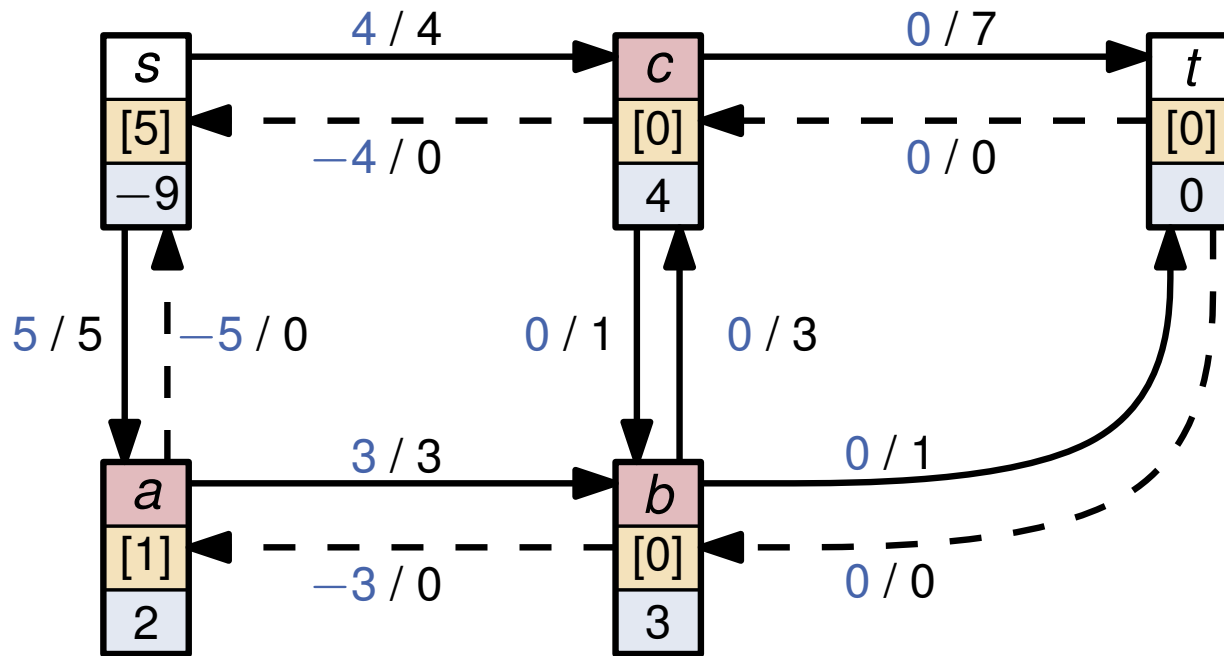


Wähle aktiven Knoten und führe PUSH / RELABEL aus:

Bedingungen zum Ausführen von PUSH für a :

- D_f enthält Kante (a, v) mit $\text{dist}(a) = \text{dist}(v) + 1$

Erhöhe Fluss auf dieser Kante (a, v) möglichst stark.

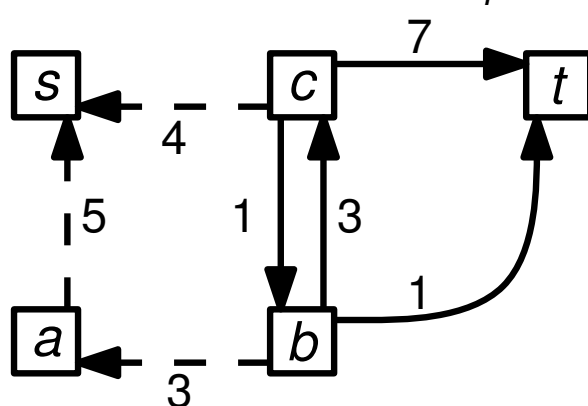


Fluss f / Kapazität c

- Führe Gegenkanten ein
- Initialisiere dist und f
- Betrachte Residualnetzwerk

Wähle aktiven Knoten und führe PUSH / RELABEL aus:

Residualnetzwerk D_f :



Bedingungen zum Ausführen von PUSH für a :

- D_f enthält Kante (a, v) mit $\text{dist}(a) = \text{dist}(v) + 1$
- Erhöhe Fluss auf dieser Kante (a, v) möglichst stark.

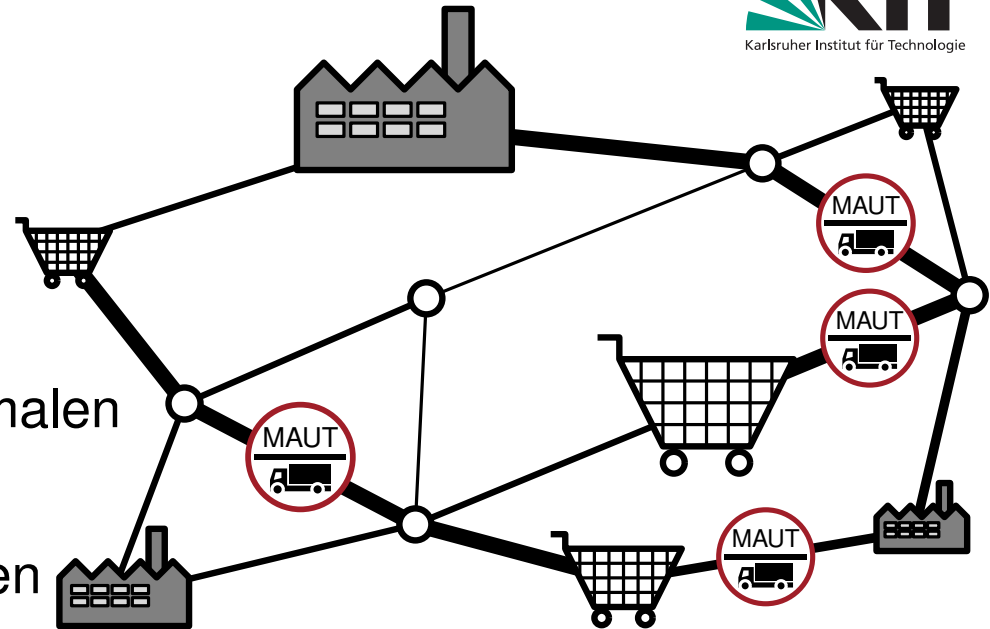
... oder auch PUSH – RELABEL

- Was ist die Idee des Goldberg-Tarjan Algos?
- Was versteht man in diesem Kontext unter „Fluss“ und „Residualnetzwerk“?
- Was ist ein Präfluss? Wann ist ein Präfluss ein Fluss?
- Was sind aktive Knoten?
- Was passiert bei den Operationen PUSH bzw. RELABEL?
- Wann sind diese Operationen zulässig?

Flüsse mit Kosten

Grundlegendes

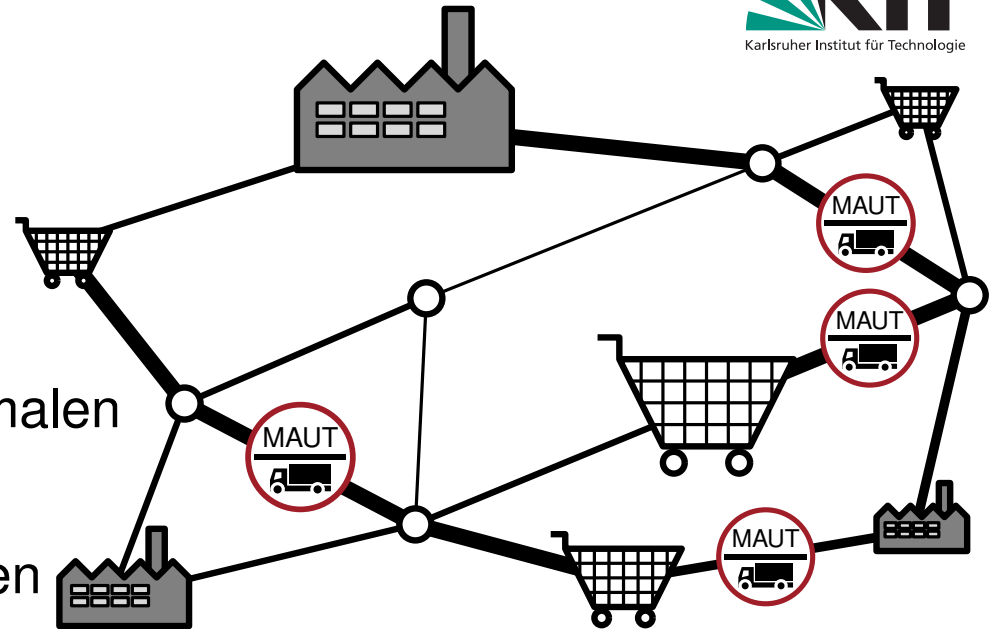
- Was ist die Problemstellung? Was sind die Kosten eines Flusses?
- Was ändert sich im Vergleich zu maximalen Flüssen?
- Was hat es mit den Quellen und Senken auf sich? Wie ändert sich die Flusserhaltungsbedingung?



Flüsse mit Kosten

Grundlegendes

- Was ist die Problemstellung? Was sind die Kosten eines Flusses?
- Was ändert sich im Vergleich zu maximalen Flüssen?
- Was hat es mit den Quellen und Senken auf sich? Wie ändert sich die Flusserhaltungsbedingung?



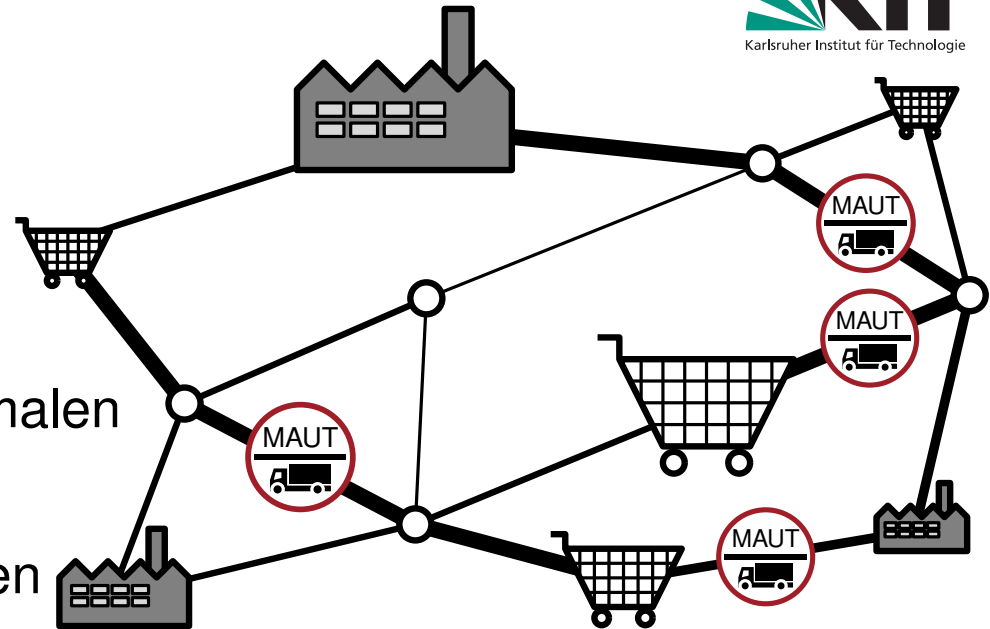
Algorithmisches

- Wie geht man mit mehreren Quellen und Senken um? Wie findet man einen gültigen Fluss?

Flüsse mit Kosten

Grundlegendes

- Was ist die Problemstellung? Was sind die Kosten eines Flusses?
- Was ändert sich im Vergleich zu maximalen Flüssen?
- Was hat es mit den Quellen und Senken auf sich? Wie ändert sich die Flusserhaltungsbedingung?

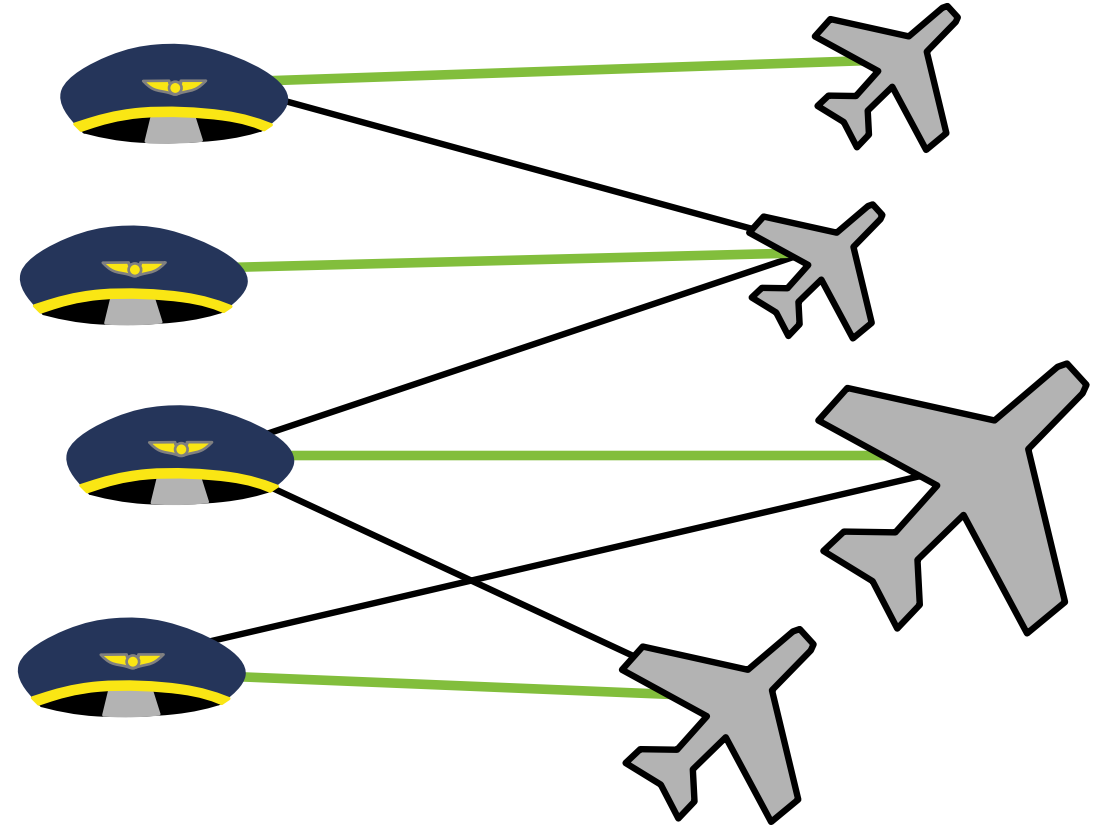
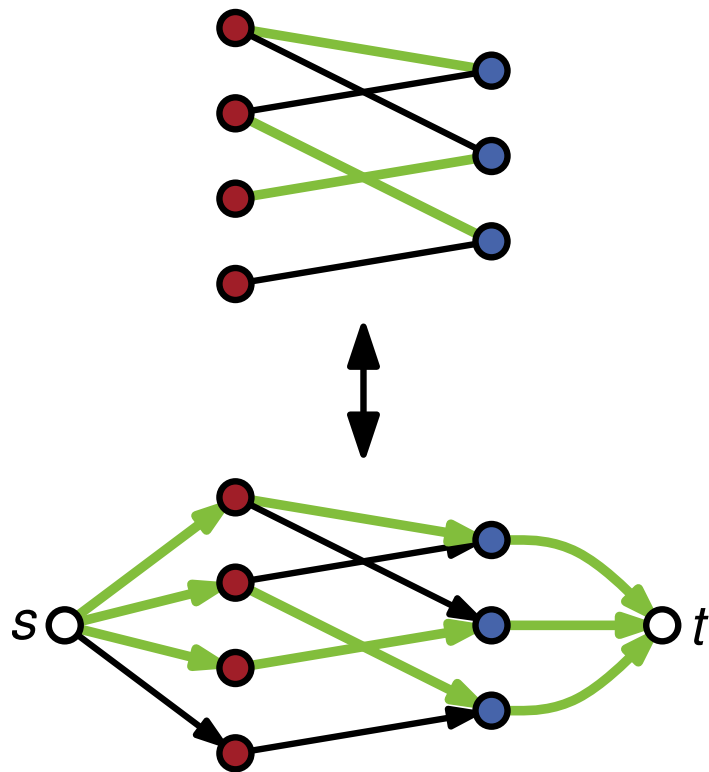


Algorithmisches

- Wie geht man mit mehreren Quellen und Senken um? Wie findet man einen gültigen Fluss?
- Was ist eine erhöhende Zirkulation? Was ist ein erhöhender Kreis?
- Wie können erhöhende Kreise zur Kostenminimierung genutzt werden?

Matchings

- Was ist ein Matching in einem Graphen?



- Was haben maximale Matchings in bipartiten Graphen mit Flüssen zu tun?

(Global) minimale Schnitte

Grundlegendes

- Was ist ein (global) minimaler Schnitt in einem Graphen?
- Wie kann man mit $n - 1$ Flussberechnungen einen minimalen Schnitt finden?
- Warum brauchen wir dann noch einen extra Algorithmus?

(Global) minimale Schnitte

Grundlegendes

- Was ist ein (global) minimaler Schnitt in einem Graphen?
- Wie kann man mit $n - 1$ Flussberechnungen einen minimalen Schnitt finden?
- Warum brauchen wir dann noch einen extra Algorithmus?

Algorithmus von Stoer & Wagner

- Wie funktioniert das?

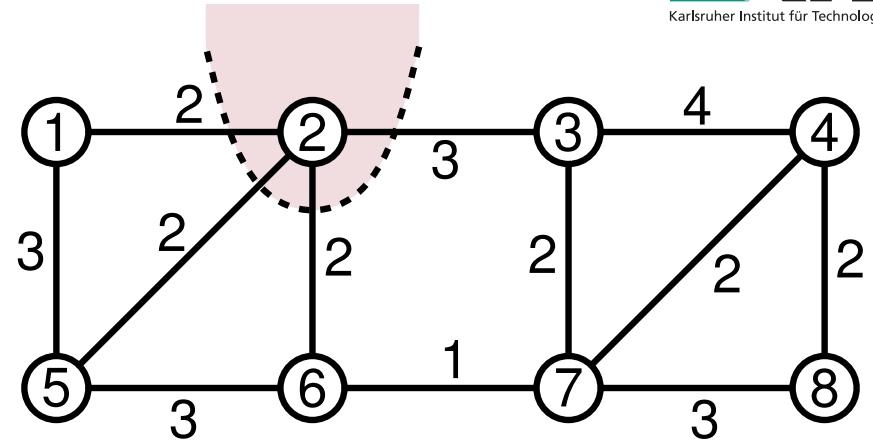
Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

$$S_1 = \{2\}$$

(beliebig gewählter Startknoten)



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

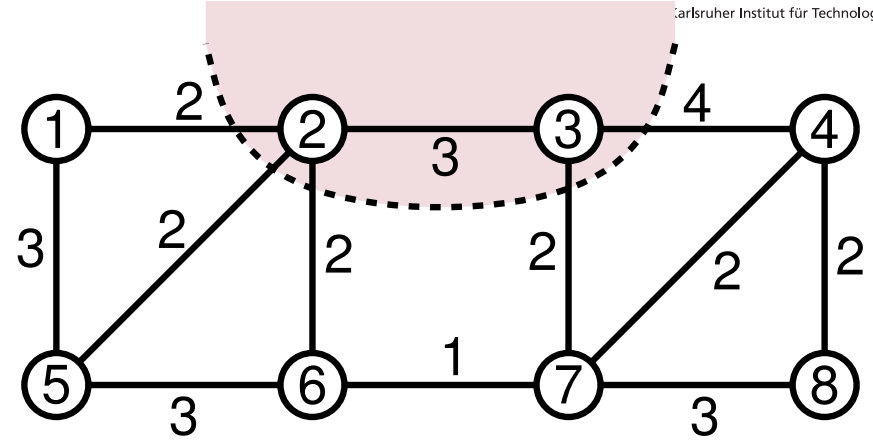
$$G_1 = G$$

$$S_1 = \{2\}$$

(beliebig gewählter Startknoten)

$$S_1 = \{2, 3\}$$

(3 am stärksten zu $\{2\}$ verbunden)



Algorithmus von Stoer & Wagner – Beispiel

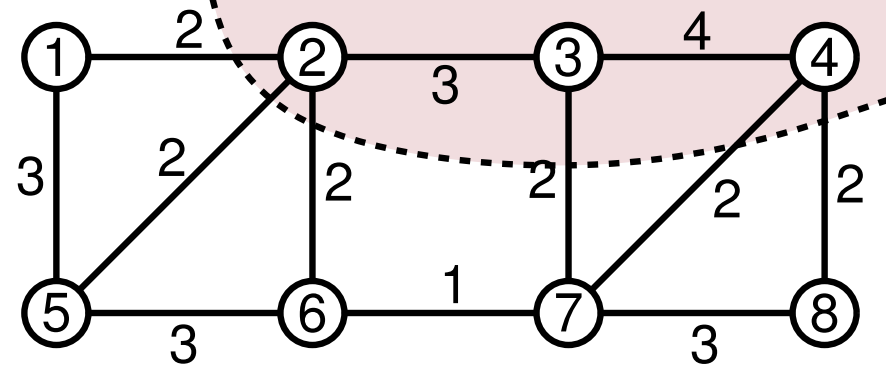
Phase 1

$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

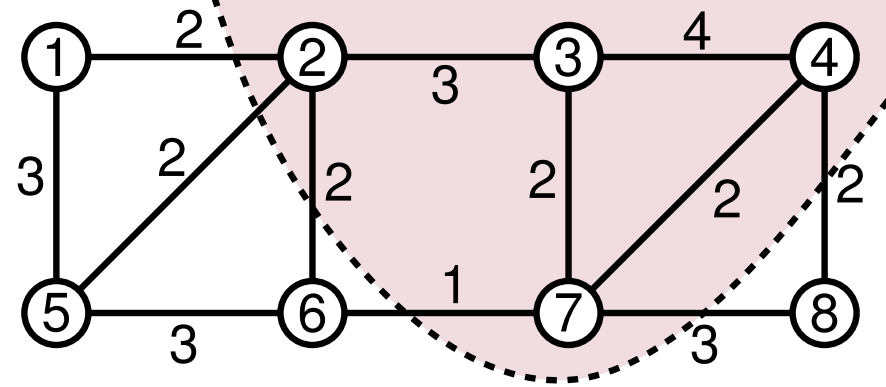
$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4, 7\}$$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

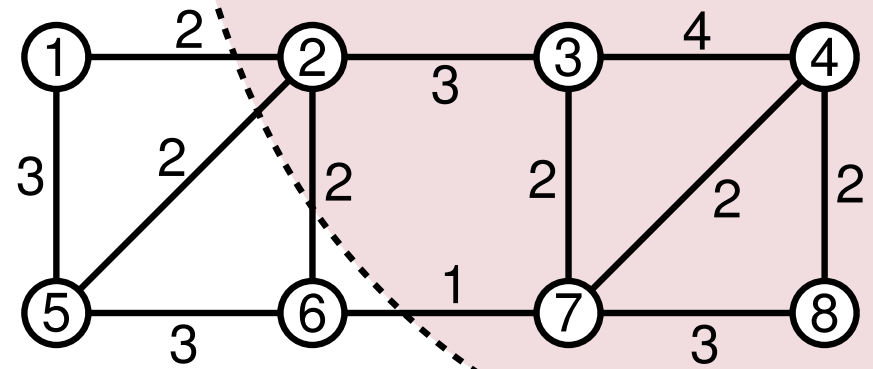
$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4, 7\}$$

$$S_1 = \{2, 3, 4, 7, 8\}$$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

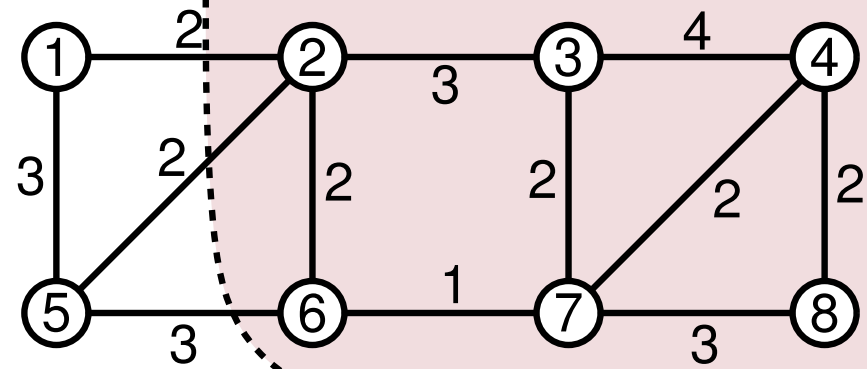
$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4, 7\}$$

$$S_1 = \{2, 3, 4, 7, 8\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6\}$$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

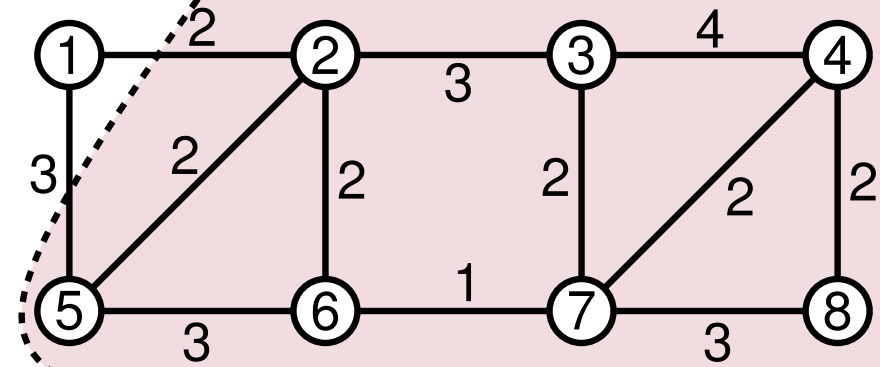
$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4, 7\}$$

$$S_1 = \{2, 3, 4, 7, 8\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

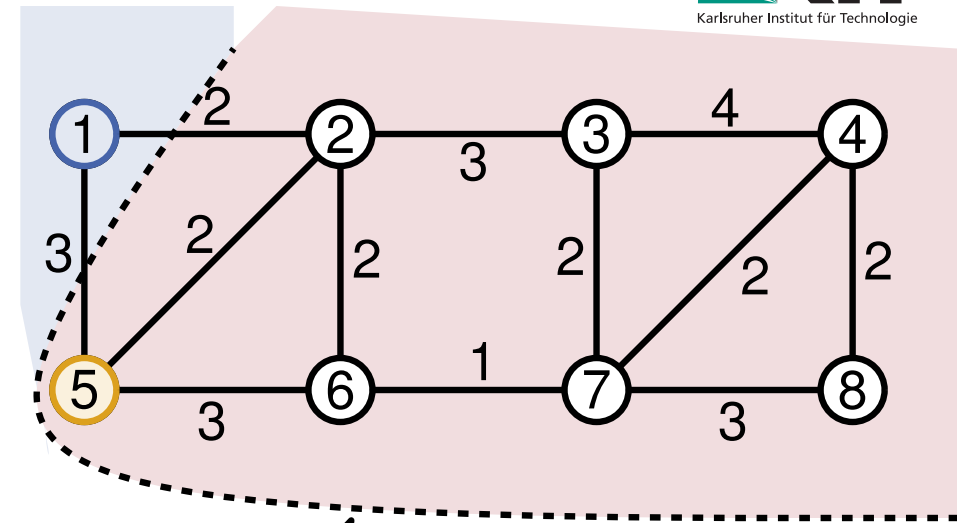
$$S_1 = \{2, 3, 4, 7\}$$

$$S_1 = \{2, 3, 4, 7, 8\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6, 5, 1\}$$



Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\}$
 \rightarrow Gewicht 5

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$$G_1 = G$$

$$S_1 = \{2\} \quad (\text{beliebig gewählter Startknoten})$$

$$S_1 = \{2, 3\} \quad (3 \text{ am stärksten zu } \{2\} \text{ verbunden})$$

$$S_1 = \{2, 3, 4\} \quad (4 \text{ am stärksten zu } \{2, 3\} \text{ verbunden})$$

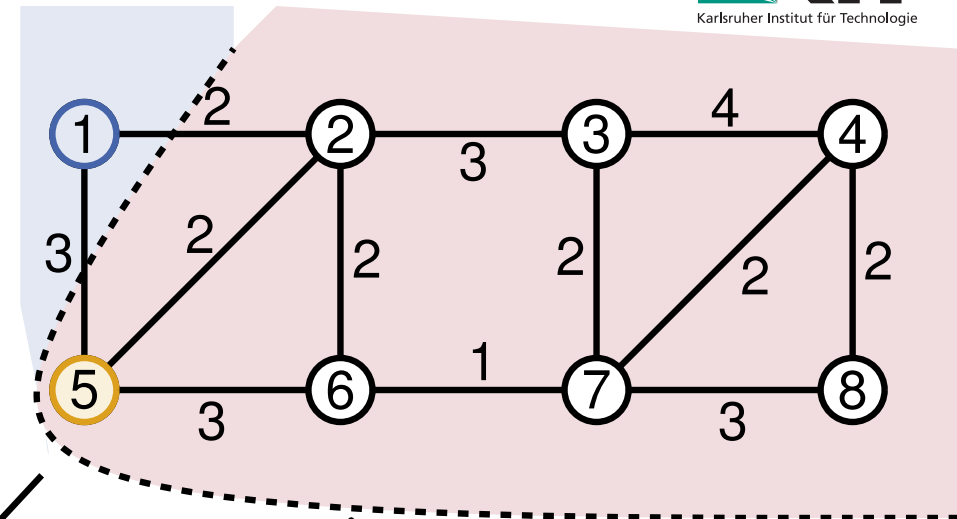
$$S_1 = \{2, 3, 4, 7\}$$

$$S_1 = \{2, 3, 4, 7, 8\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6\}$$

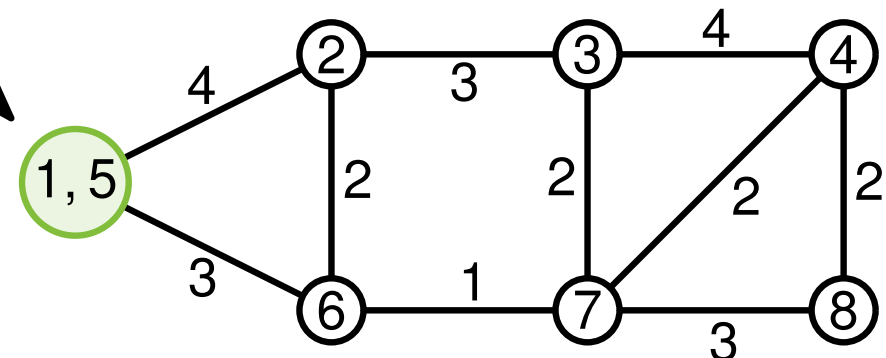
$$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$$

$$S_1 = \{2, 3, 4, 7, 8, 6, 5, 1\}$$



Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\}$
 \rightarrow Gewicht 5

Verschmelzen von s und t ergibt G_2



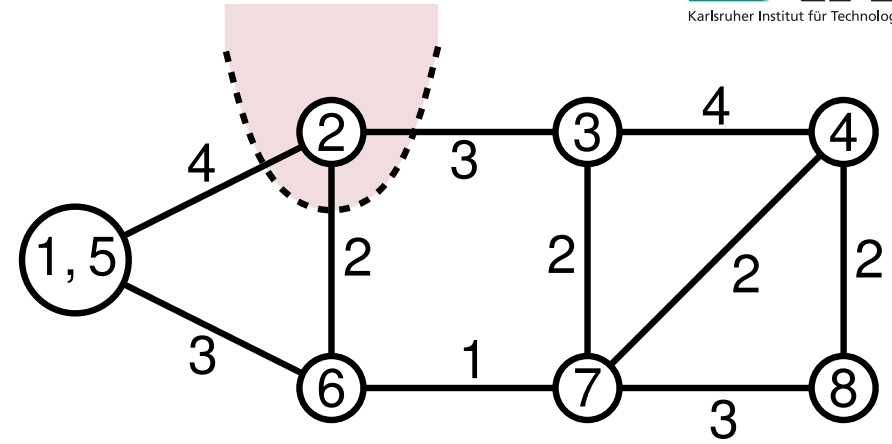
Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$

(beliebig gewählter Startknoten)



Algorithmus von Stoer & Wagner – Beispiel

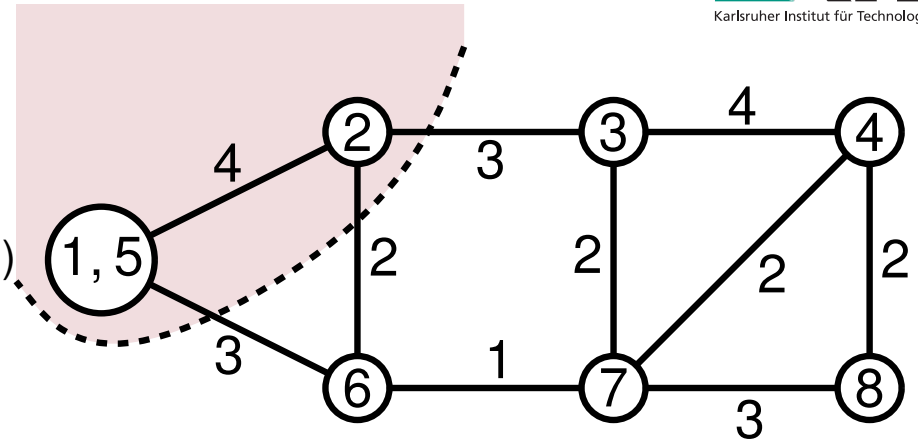
Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$$S_2 = \{2\}$$

(beliebig gewählter Startknoten)

$$S_2 = \{2, \{1, 5\}\}$$



Algorithmus von Stoer & Wagner – Beispiel

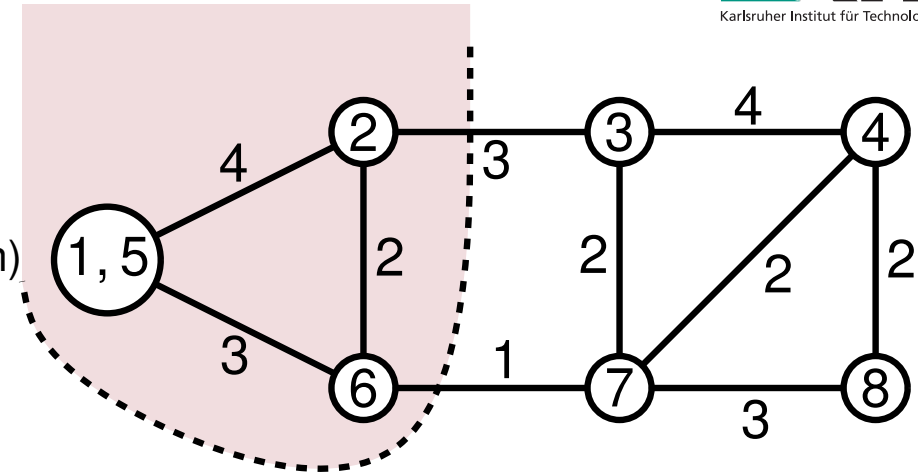
Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 2

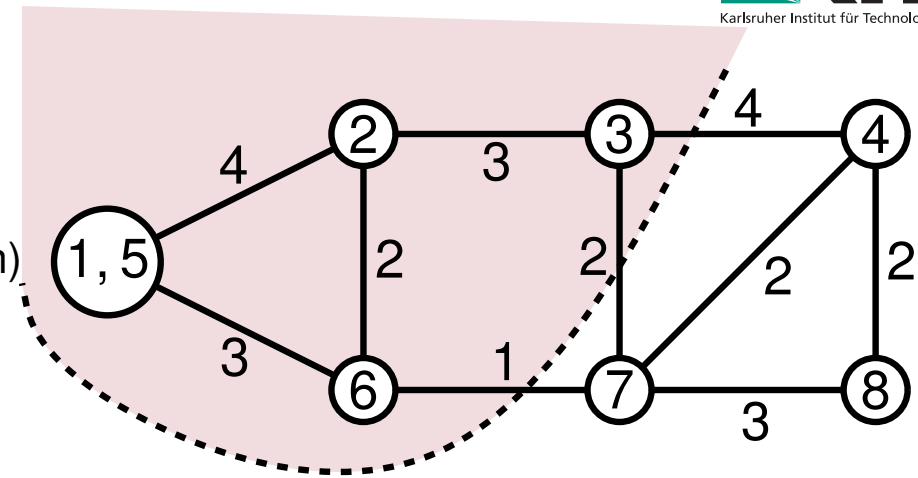
$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

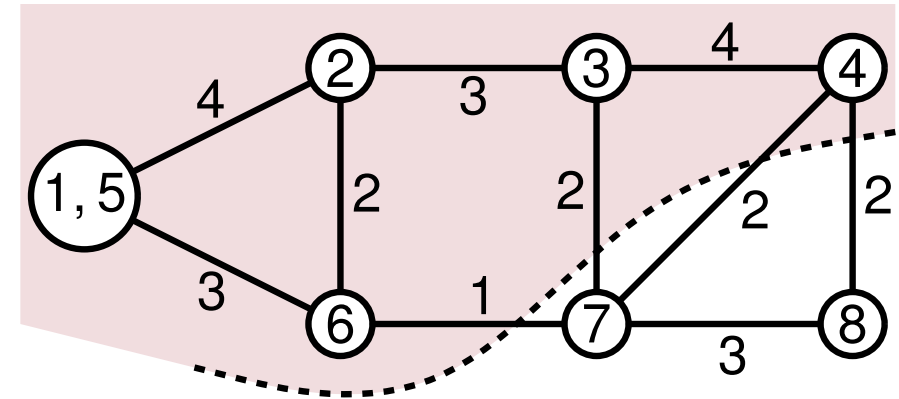
$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

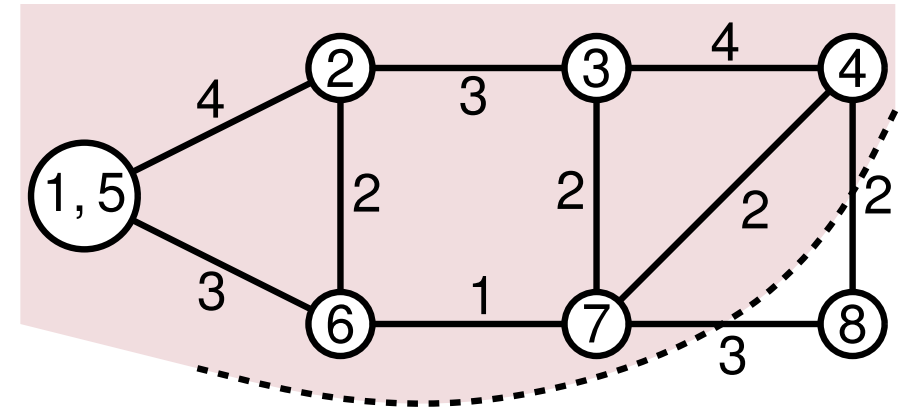
$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4, 7\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$

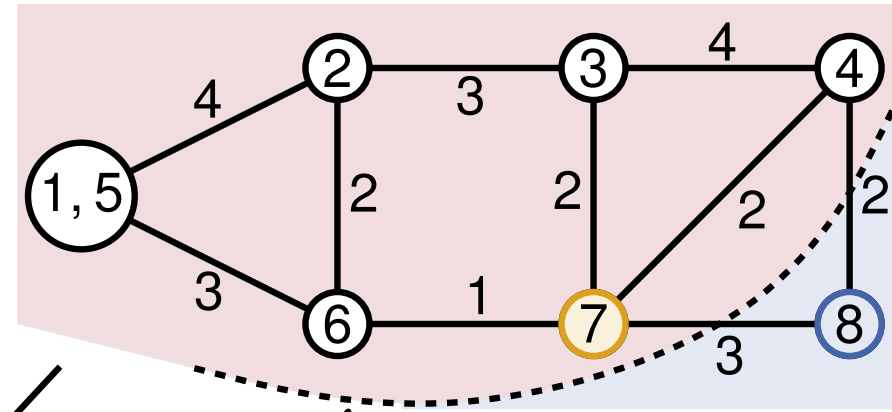
$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4\}$

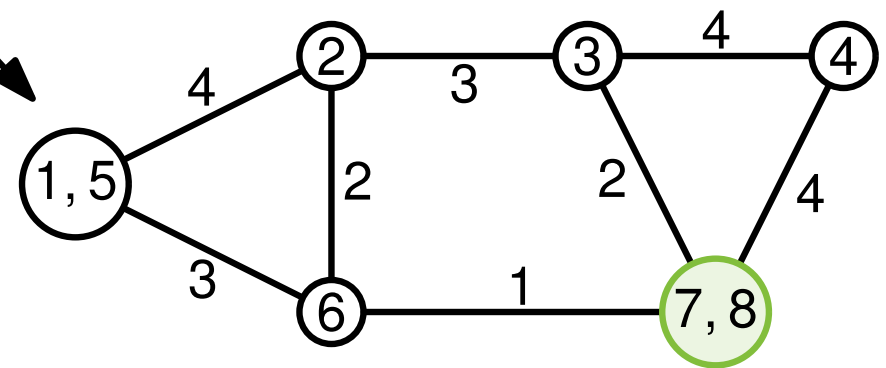
$S_2 = \{2, \{1, 5\}, 6, 3, 4, 7\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4, 7, 8\}$



Schnitt der Phase: $\{V_2 \setminus \{8\}, \{8\}\}$
 → Gewicht 5

Verschmelzen von s und t ergibt G_3



Algorithmus von Stoer & Wagner – Beispiel

Phase 1 Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\} \rightarrow$ Gewicht 5

Phase 2 Schnitt der Phase: $\{V_2 \setminus \{8\}, \{8\}\} \rightarrow$ Gewicht 5

Phase 3 Schnitt der Phase: $\{V_3 \setminus \{\{7, 8\}\}, \{\{7, 8\}\}\} \rightarrow$ Gewicht 7

Phase 4 Schnitt der Phase: $\{V_4 \setminus \{\{4, 7, 8\}\}, \{\{4, 7, 8\}\}\} \rightarrow$ Gewicht 7

Phase 5 Schnitt der Phase: $\{V_5 \setminus \{\{3, 4, 7, 8\}\}, \{\{3, 4, 7, 8\}\}\} \rightarrow$ Gewicht 4

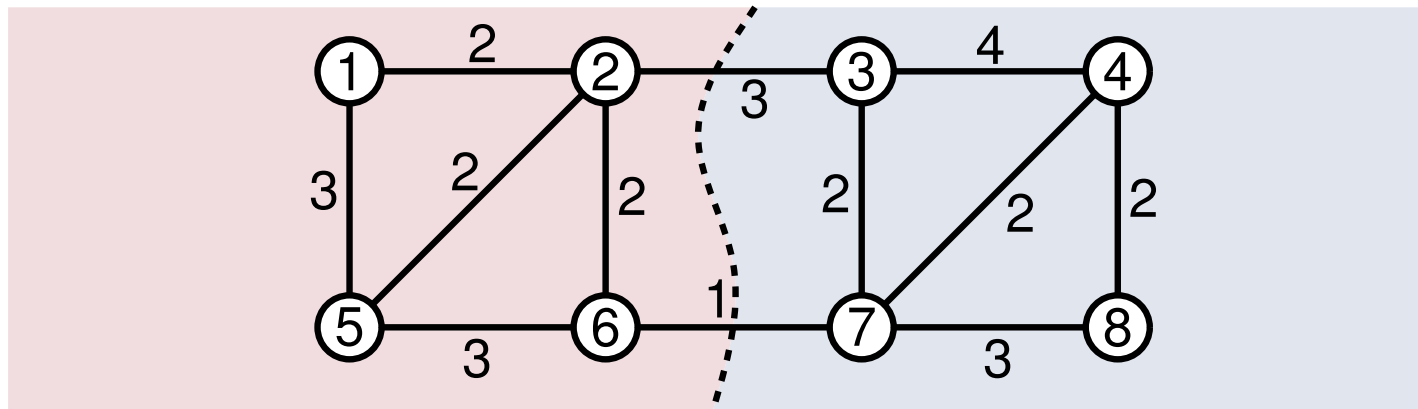
Phase 6 Schnitt der Phase: $\{V_6 \setminus \{\{1, 5\}\}, \{\{1, 5\}\}\} \rightarrow$ Gewicht 7

Phase 7 Schnitt der Phase: $\{V_7 \setminus \{2\}, \{2\}\} \rightarrow$ Gewicht 9

] siehe Skript

Der Schnitt aus **Phase 5** ist minimal unter den Schnitten der einzelnen Phasen.

\Rightarrow Der Algorithmus von Stoer & Wagner gibt diesen Schnitt aus.



(Beweis, dass der so bestimmte Schnitt immer ein minimaler Schnitt ist folgt später.)

(Global) minimale Schnitte

Grundlegendes

- Was ist ein (global) minimaler Schnitt in einem Graphen?
- Wie kann man mit $n - 1$ Flussberechnungen einen minimalen Schnitt finden?
- Warum brauchen wir dann noch einen extra Algorithmus?

Algorithmus von Stoer & Wagner

- Wie funktioniert das?

(Global) minimale Schnitte

Grundlegendes

- Was ist ein (global) minimaler Schnitt in einem Graphen?
- Wie kann man mit $n - 1$ Flussberechnungen einen minimalen Schnitt finden?
- Warum brauchen wir dann noch einen extra Algorithmus?

Algorithmus von Stoer & Wagner

- Wie funktioniert das?
- Was heißt „am stärksten verbunden“?
- Wie werden Knoten verschmolzen?

(Global) minimale Schnitte

Grundlegendes

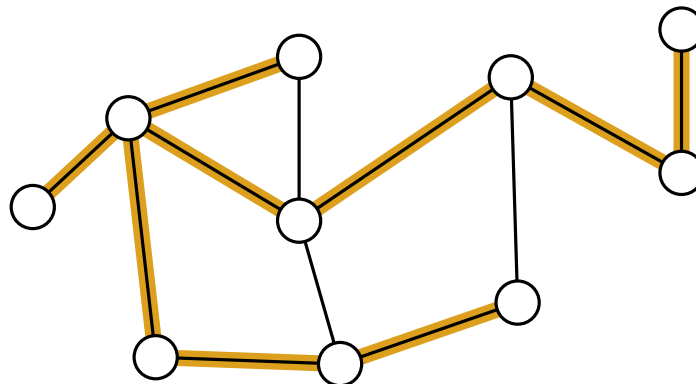
- Was ist ein (global) minimaler Schnitt in einem Graphen?
- Wie kann man mit $n - 1$ Flussberechnungen einen minimalen Schnitt finden?
- Warum brauchen wir dann noch einen extra Algorithmus?

Algorithmus von Stoer & Wagner

- Wie funktioniert das?
- Was heißt „am stärksten verbunden“?
- Wie werden Knoten verschmolzen?
- Was hat es mit s und t und dem Schnitt jeder Phase auf sich?
- Warum funktioniert der Algorithmus?

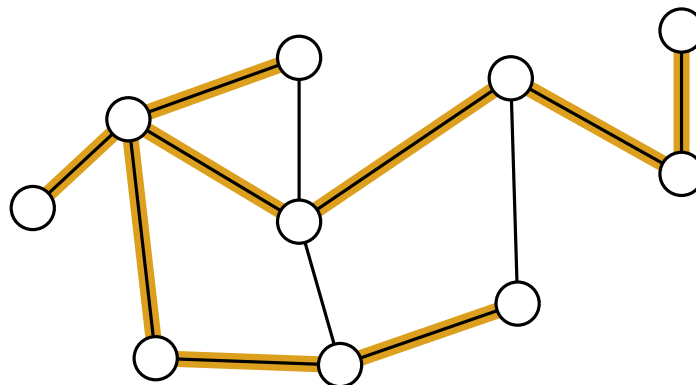
- Was ist ein Unabhängigkeitssystem?
- Wann ist ein Unabhängigkeitssystem ein Matroid?

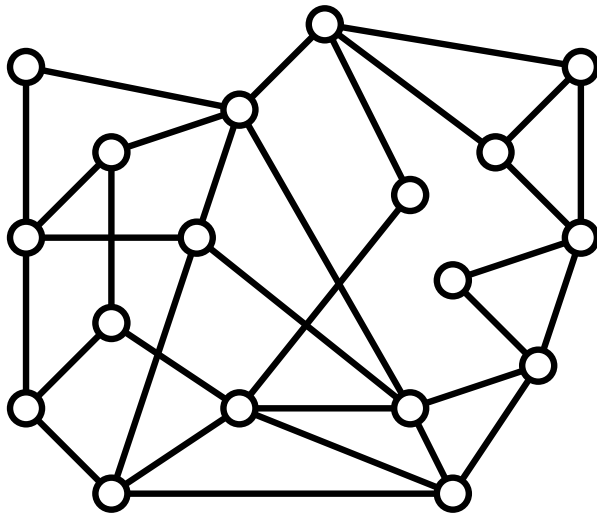
Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem (E, \mathcal{U}) mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.



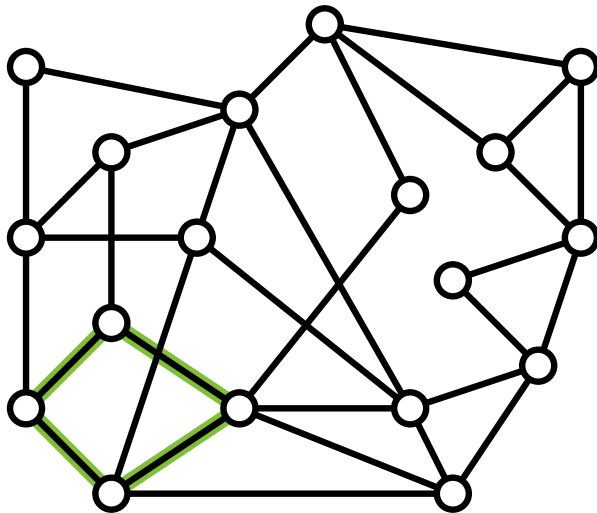
- Was ist ein Unabhängigkeitssystem?
- Wann ist ein Unabhängigkeitssystem ein Matroid?
- Was ist eine Basis eines Unabhängigkeitssystems bzw. Matroids?
- Wie kann man in Matroiden maximale unabhängige Mengen, bzw. minimale Basen finden?
- Wie funktioniert die Greedy-Methode?

Beispiel: Sei $G = (V, E)$ ein zusammenhängender Graph. Mengensystem (E, \mathcal{U}) mit $\mathcal{U} = \{E' \subseteq E : E' \text{ induziert einen Wald in } G\}$ ist ein Unabhängigkeitssystem.

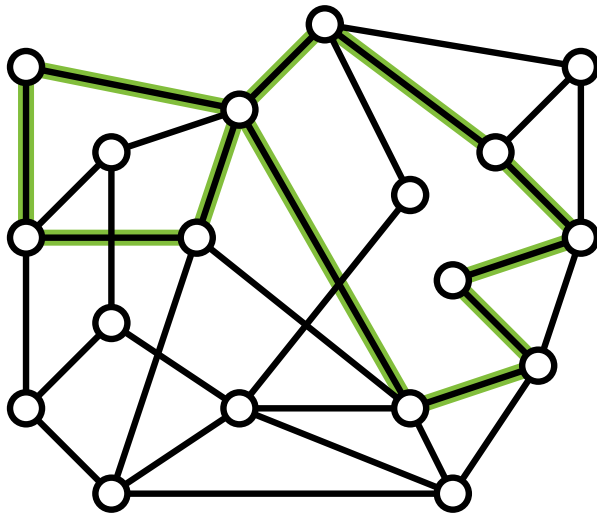




- Ein Graph kann sehr viele Kreise haben.

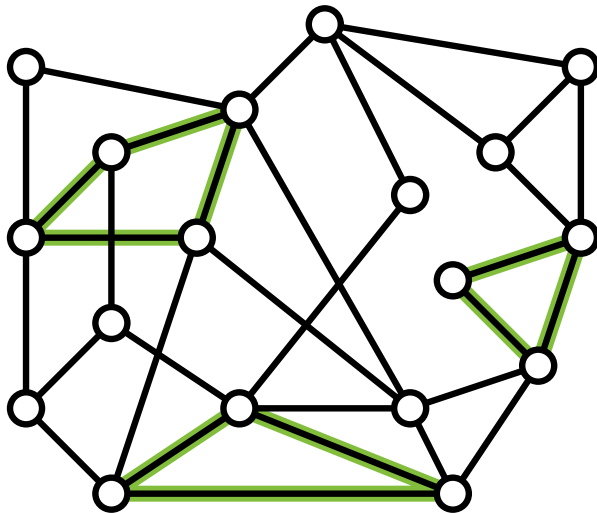


- Ein Graph kann sehr viele Kreise haben.

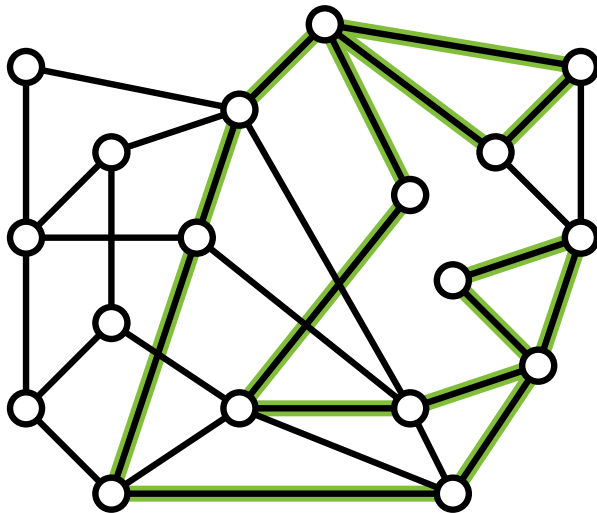


- Ein Graph kann sehr viele Kreise haben.

Kreisbasen

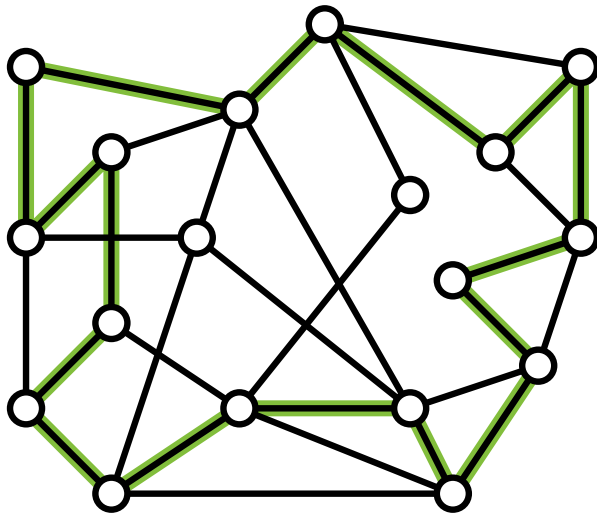


- Ein Graph kann sehr viele Kreise haben.

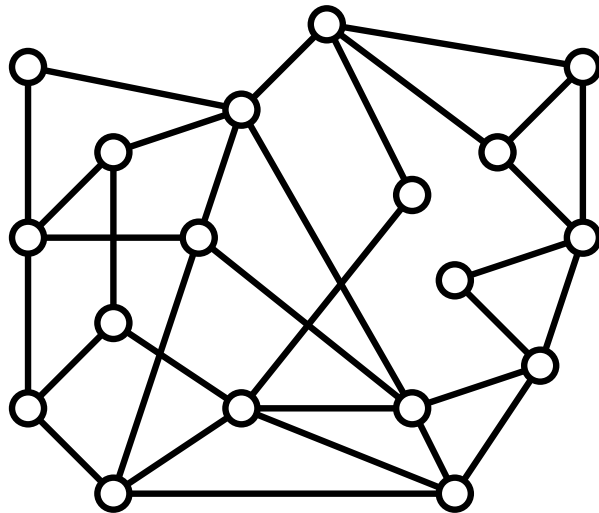


- Ein Graph kann sehr viele Kreise haben.

Kreisbasen

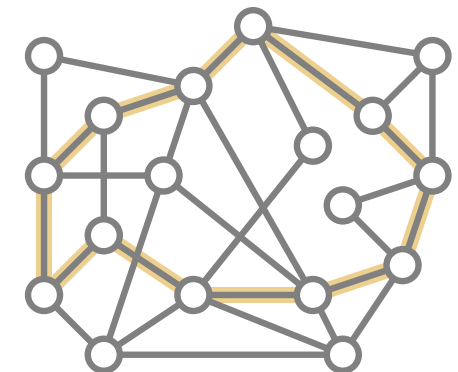
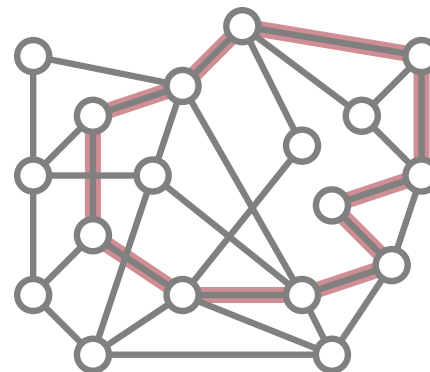
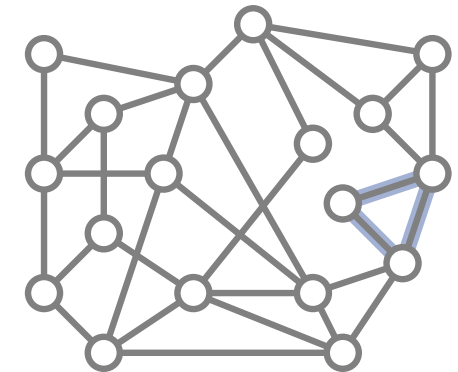
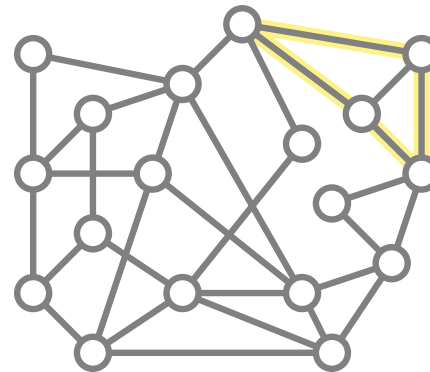


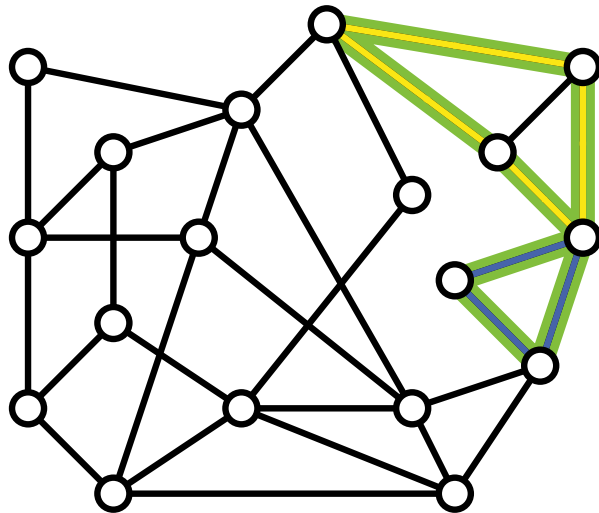
- Ein Graph kann sehr viele Kreise haben.



- Ein Graph kann sehr viele Kreise haben.

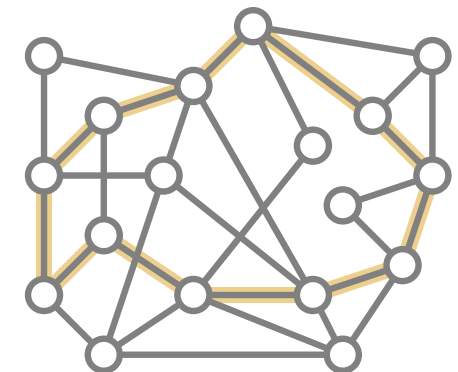
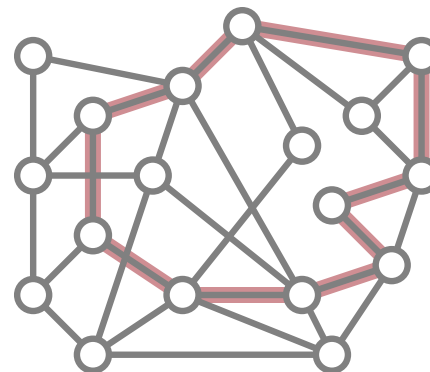
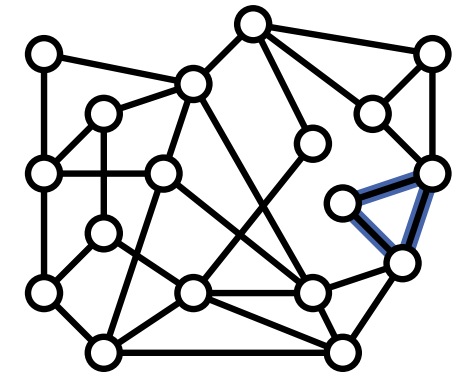
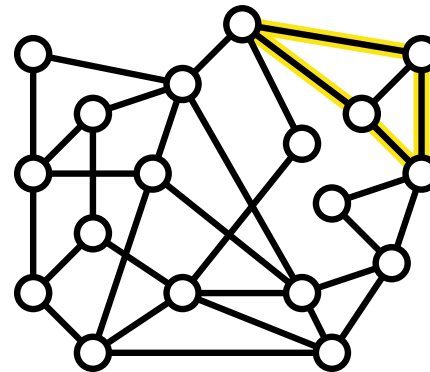
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

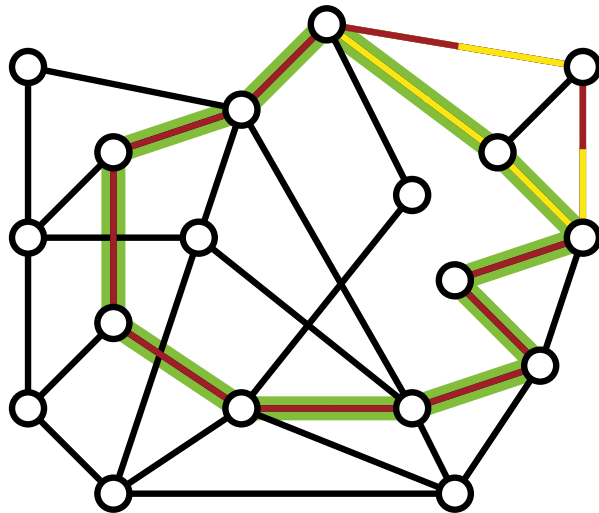




- Ein Graph kann sehr viele Kreise haben.

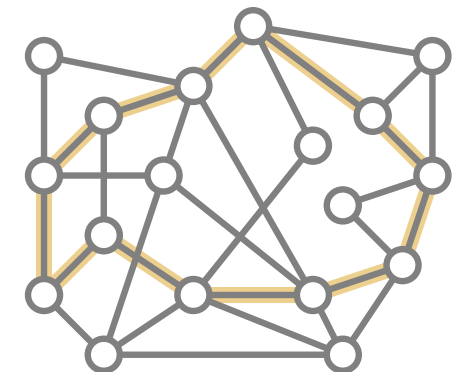
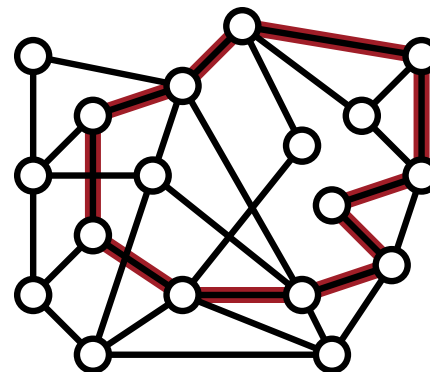
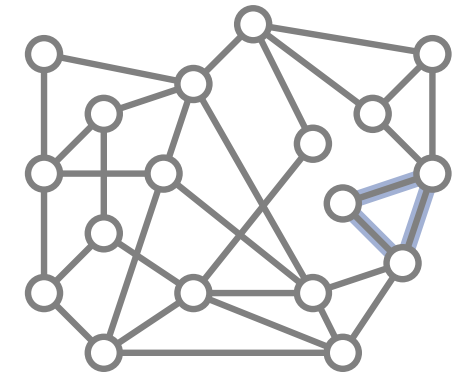
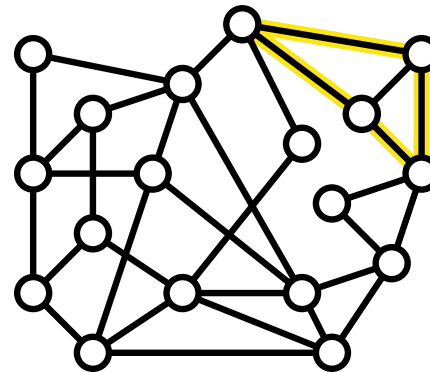
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

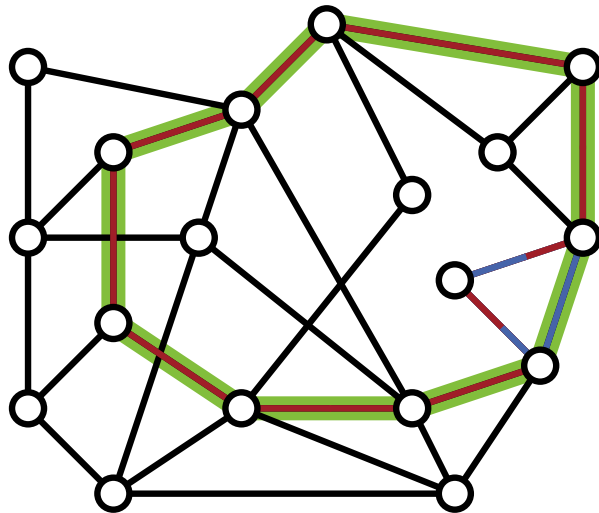




- Ein Graph kann sehr viele Kreise haben.

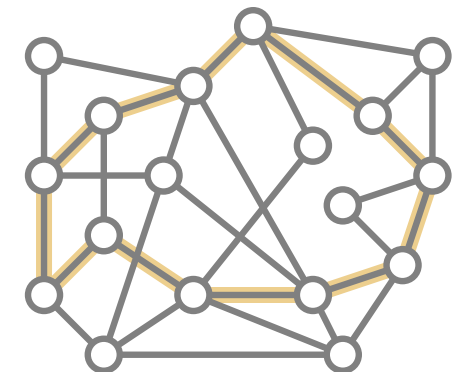
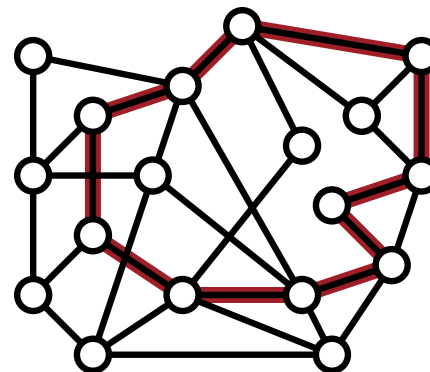
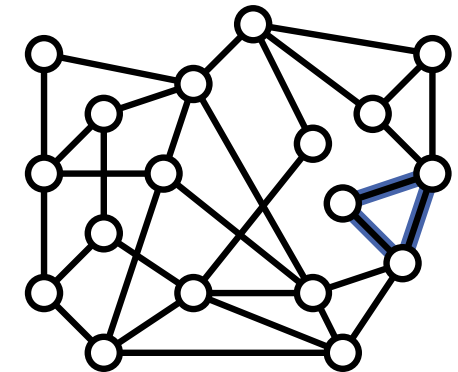
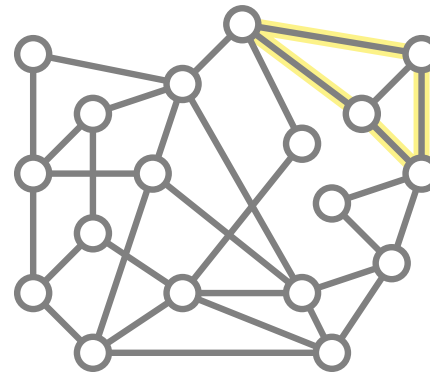
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

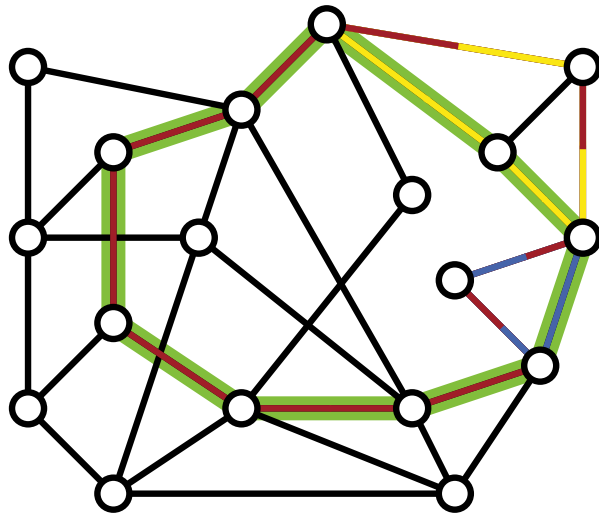




- Ein Graph kann sehr viele Kreise haben.

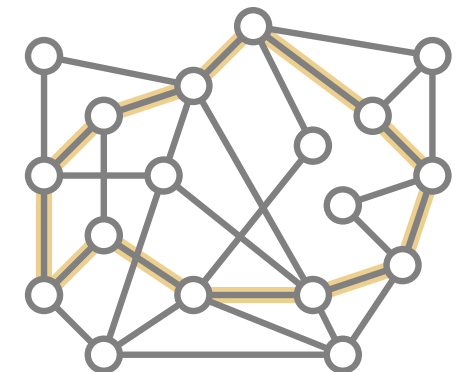
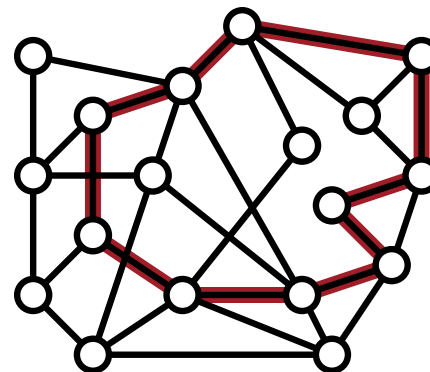
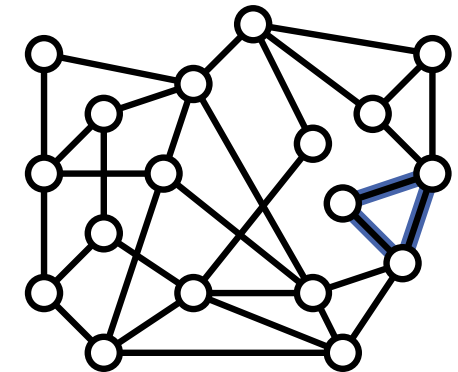
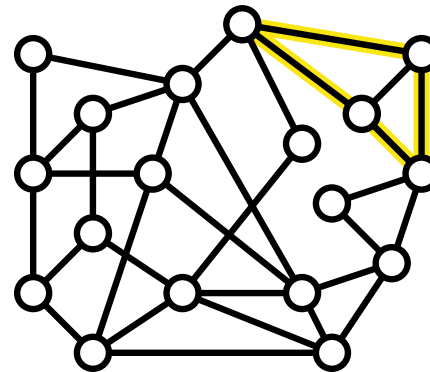
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

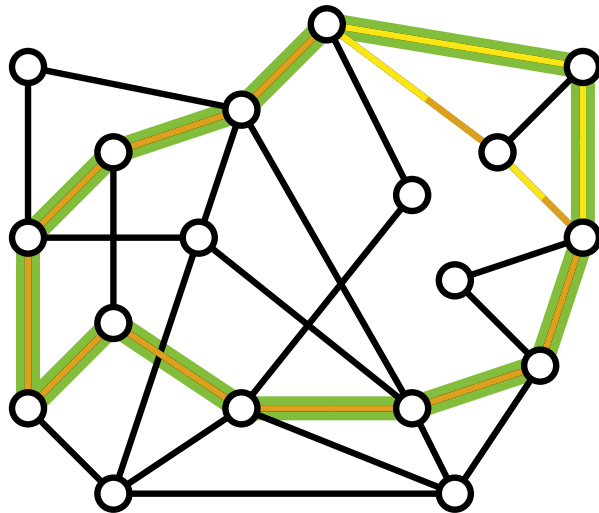




- Ein Graph kann sehr viele Kreise haben.

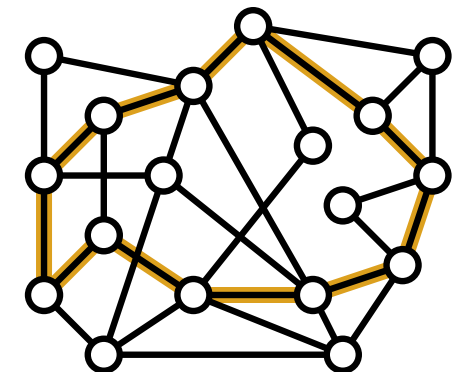
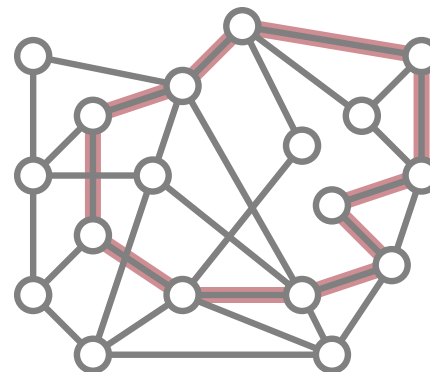
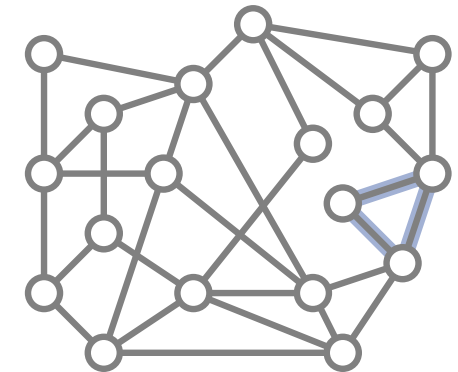
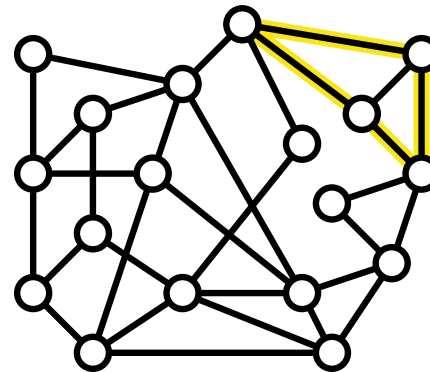
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

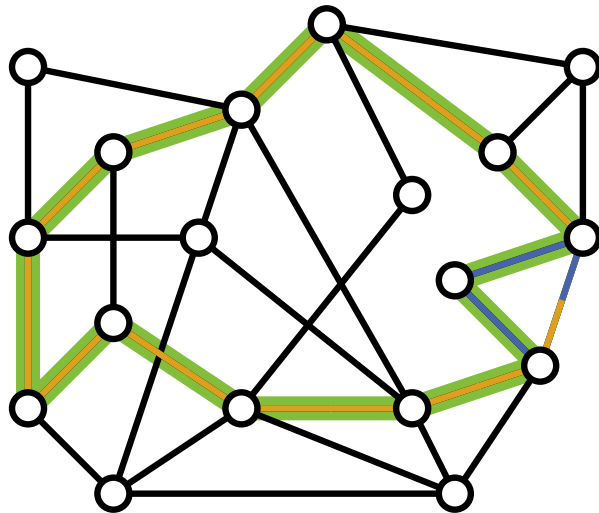




- Ein Graph kann sehr viele Kreise haben.

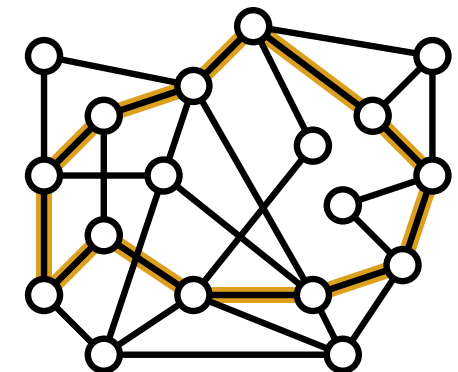
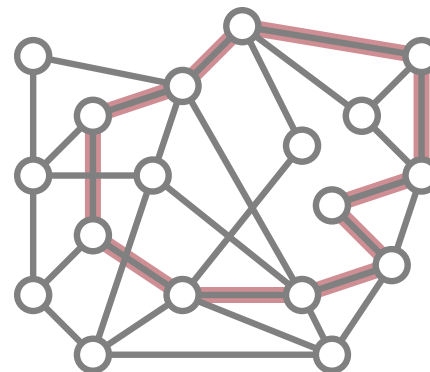
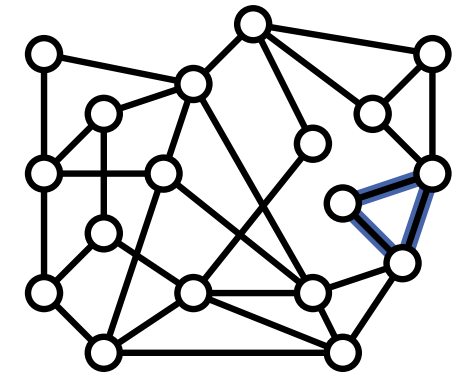
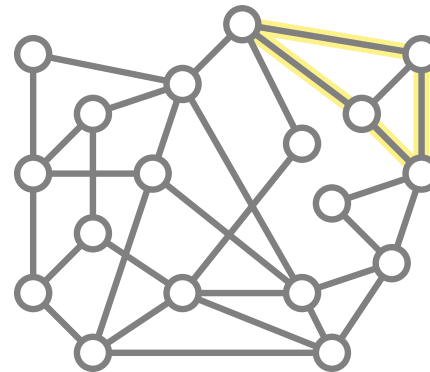
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

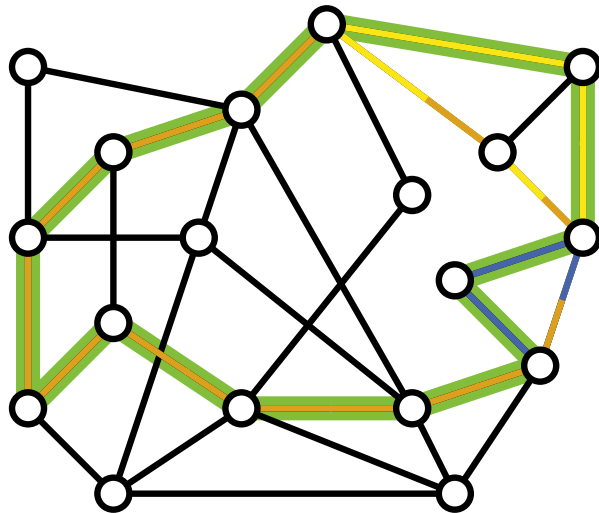




- Ein Graph kann sehr viele Kreise haben.

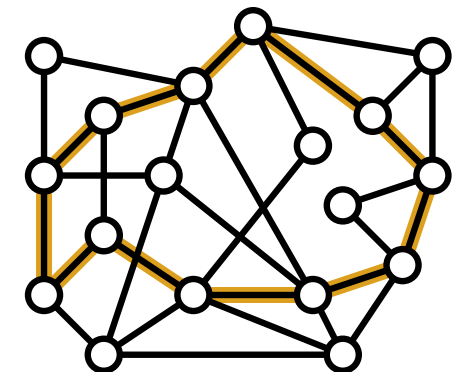
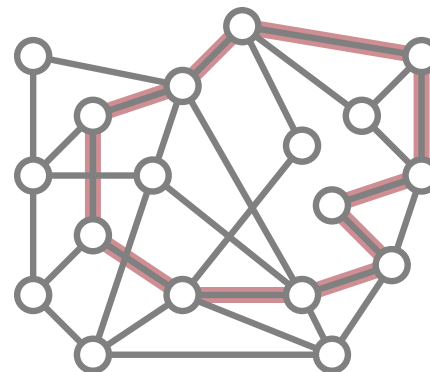
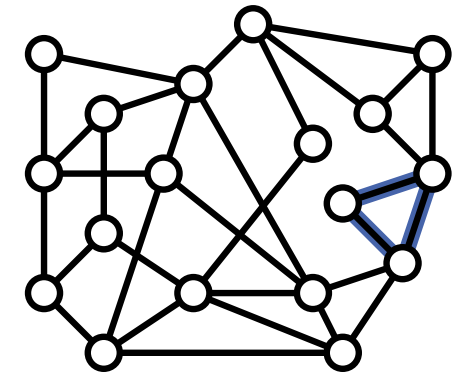
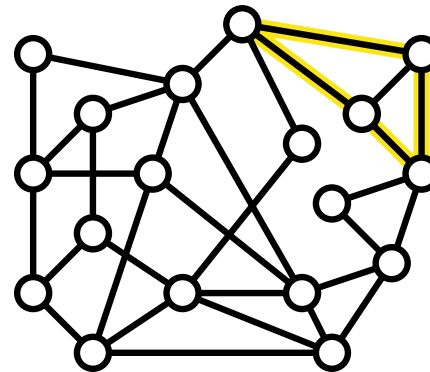
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

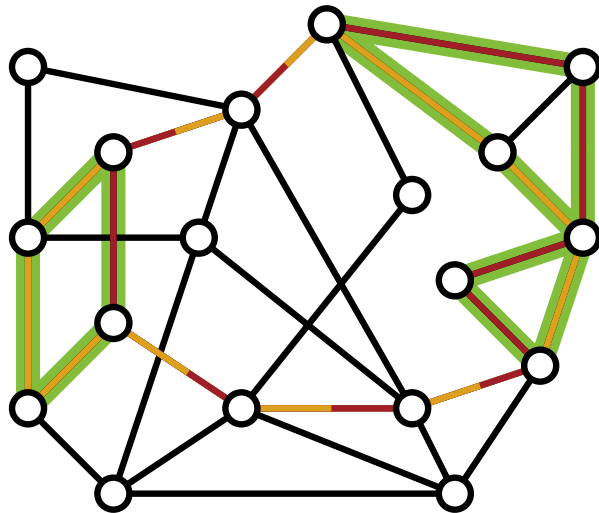




- Ein Graph kann sehr viele Kreise haben.

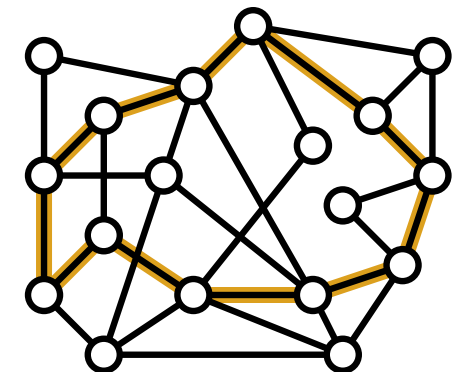
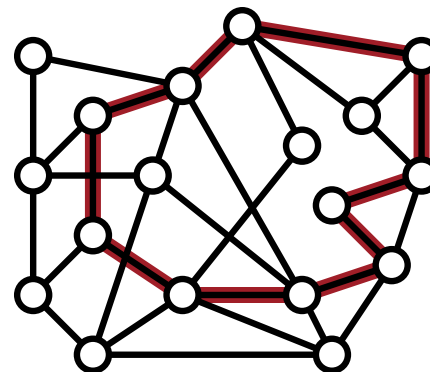
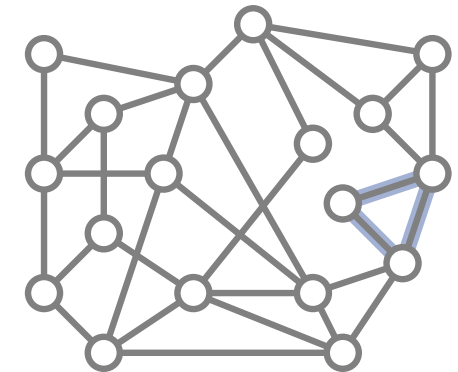
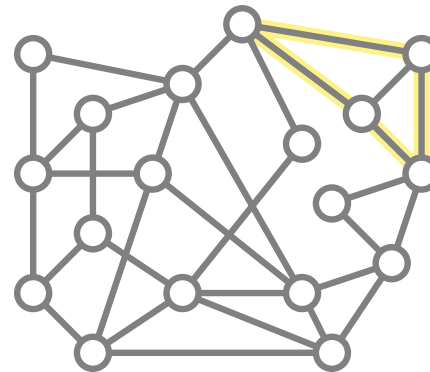
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

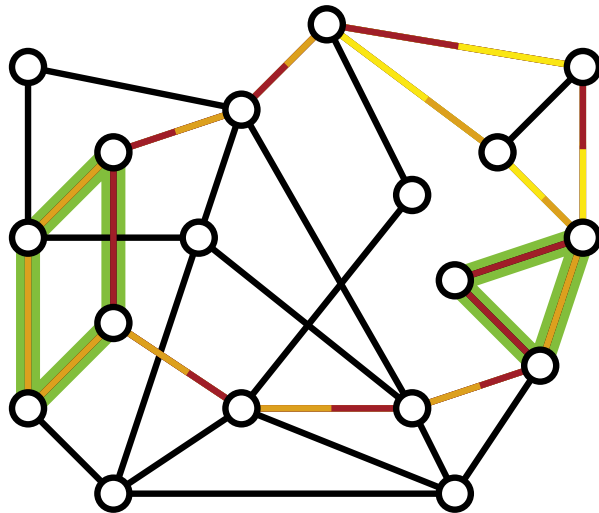




- Ein Graph kann sehr viele Kreise haben.

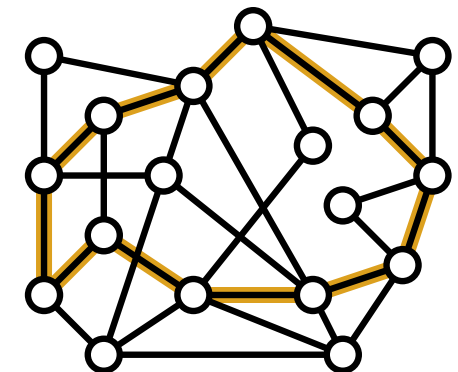
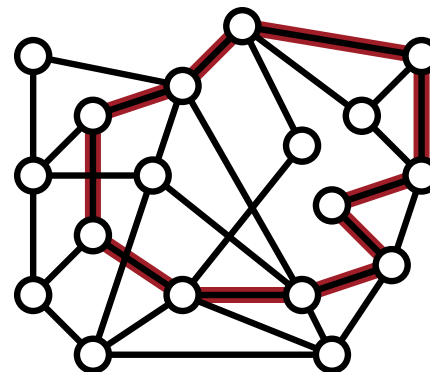
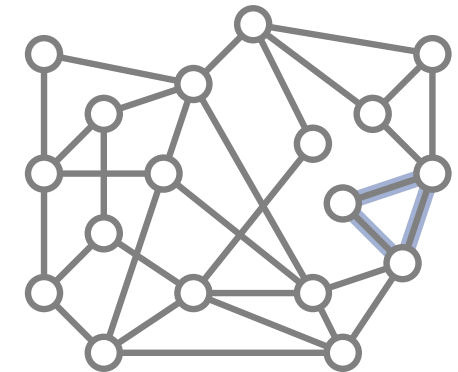
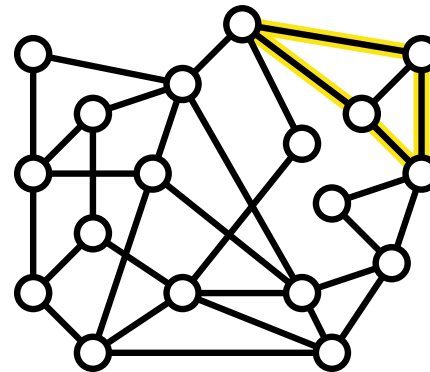
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

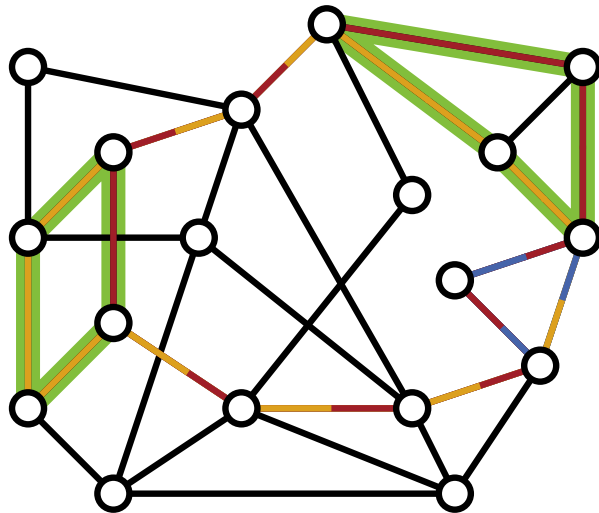




- Ein Graph kann sehr viele Kreise haben.

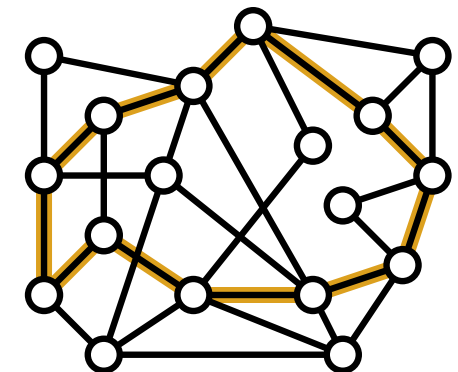
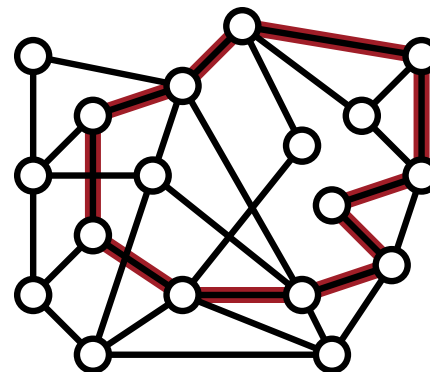
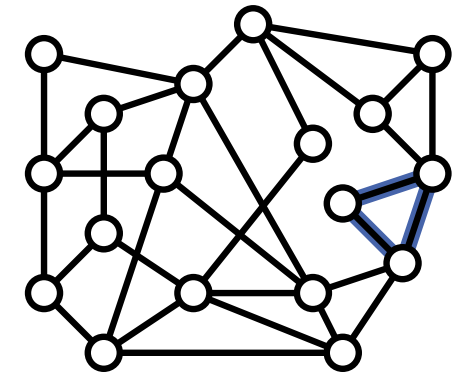
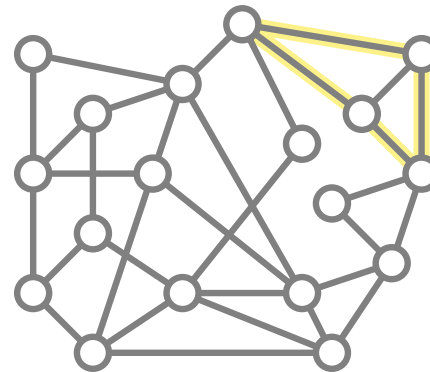
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?

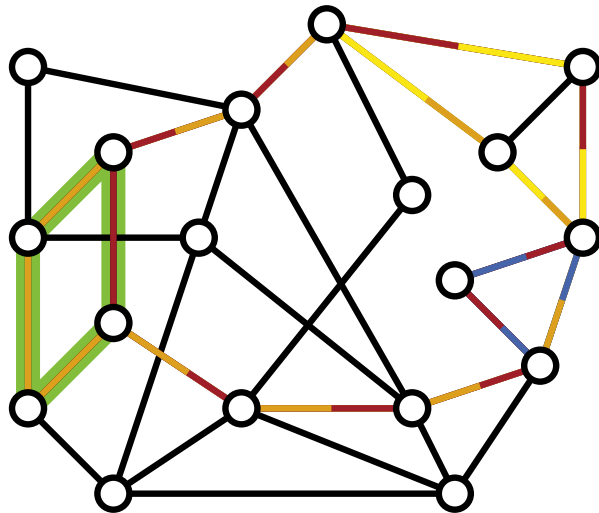




- Ein Graph kann sehr viele Kreise haben.

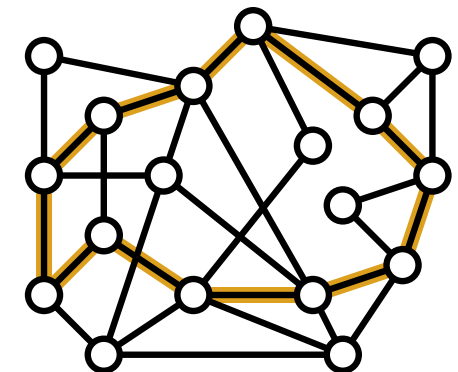
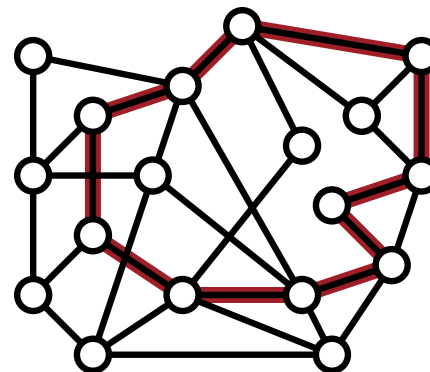
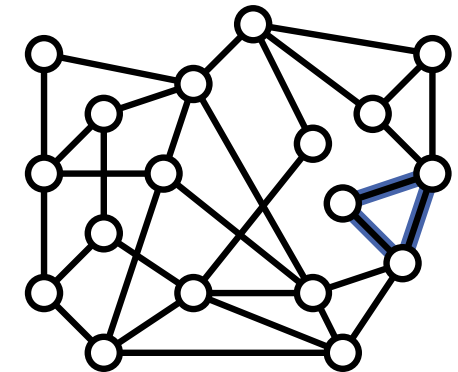
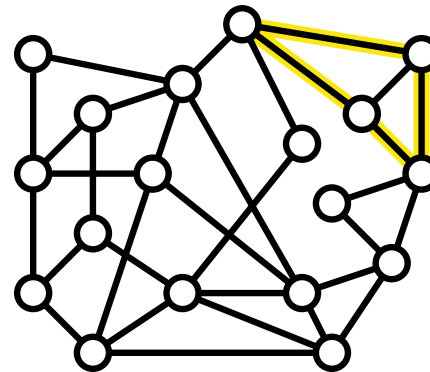
- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?





- Ein Graph kann sehr viele Kreise haben.

- Man kann aus wenigen Kreisen viele zusammensetzen.
- Wie viele braucht man, um alle Kreise zu erzeugen?



Grundlegendes

- Was ist ein Kreis im Kontext von Kreisräumen?
- Wie ist die Addition auf Kreisen definiert und warum liefert die Menge aller Kreise einen Vektorraum?

Grundlegendes

- Was ist ein Kreis im Kontext von Kreisräumen?
- Wie ist die Addition auf Kreisen definiert und warum liefert die Menge aller Kreise einen Vektorraum?
- Was ist ein Fundamentalkreis? Was ist eine Fundamentalbasis des Kreisraums? Ist jede Kreisbasis eine Fundamentalbasis?
- Was ist die Dimension eines Kreisraums?

Grundlegendes

- Was ist ein Kreis im Kontext von Kreisräumen?
- Wie ist die Addition auf Kreisen definiert und warum liefert die Menge aller Kreise einen Vektorraum?
- Was ist ein Fundamentalkreis? Was ist eine Fundamentalbasis des Kreisraums? Ist jede Kreisbasis eine Fundamentalbasis?
- Was ist die Dimension eines Kreisraums?

Algorithmen zur Berechnung einer Kreisbasis minimalen Gewichts

- Was haben Kreisräume mit Matroiden zu tun?
- Warum benutzt man nicht einfach die Greedy-Methode zur Berechnung einer minimalen Kreisbasis?
- Was machen die Algorithmen von Horton bzw. de Pina? Wie schnell sind sie?

Grundlegendes

- Wann ist ein Algorithmus randomisiert?
- Was unterscheidet einen Las Vegas von einem Monte Carlo Algorithmus?
- Was ist ein beidseitiger bzw. einseitiger Fehler?
- Wie sind die Probleme MAX SAT und MAX CUT definiert?

Grundlegendes

- Wann ist ein Algorithmus randomisiert?
- Was unterscheidet einen Las Vegas von einem Monte Carlo Algorithmus?
- Was ist ein beidseitiger bzw. einseitiger Fehler?
- Wie sind die Probleme MAX SAT und MAX CUT definiert?

Algorithmisches

- Wie kann man die Fehlerwahrscheinlichkeit bei einem Monte Carlo Algorithmus analysieren (Beispiel MINCUT) und wie kann man das ggf. verbessern?

Grundlegendes

- Wann ist ein Algorithmus randomisiert?
- Was unterscheidet einen Las Vegas von einem Monte Carlo Algorithmus?
- Was ist ein beidseitiger bzw. einseitiger Fehler?
- Wie sind die Probleme MAX SAT und MAX CUT definiert?

Algorithmisches

- Wie kann man die Fehlerwahrscheinlichkeit bei einem Monte Carlo Algorithmus analysieren (Beispiel MINCUT) und wie kann man das ggf. verbessern?
- Warum hat RANDOM SAT die erwartete relative Approximationsgüte 2?

Grundlegendes

- Wann ist ein Algorithmus randomisiert?
- Was unterscheidet einen Las Vegas von einem Monte Carlo Algorithmus?
- Was ist ein beidseitiger bzw. einseitiger Fehler?
- Wie sind die Probleme MAX SAT und MAX CUT definiert?

Algorithmisches

- Wie kann man die Fehlerwahrscheinlichkeit bei einem Monte Carlo Algorithmus analysieren (Beispiel MINCUT) und wie kann man das ggf. verbessern?
- Warum hat RANDOM SAT die erwartete relative Approximationsgüte 2?
- Wie kann MAX CUT als quadratisches Programm formuliert werden?
- Wie kann MAX CUT mithilfe des quadratischen Programms und mithilfe randomisierten Rundens berechnet werden.
- Liefert dieses Vorgehen immer einen maximalen Schnitt?

Grundlegendes

- Was bedeuten die Begriffe Strecke, (einfaches/konvexes) Polygon, konvexe Hülle, umschließendes Rechteck?
- Wie testet man, auf welcher Seite einer Gerade ein Punkt liegt?
- Wie testet man, ob zwei Strecken sich schneiden?

Grundlegendes

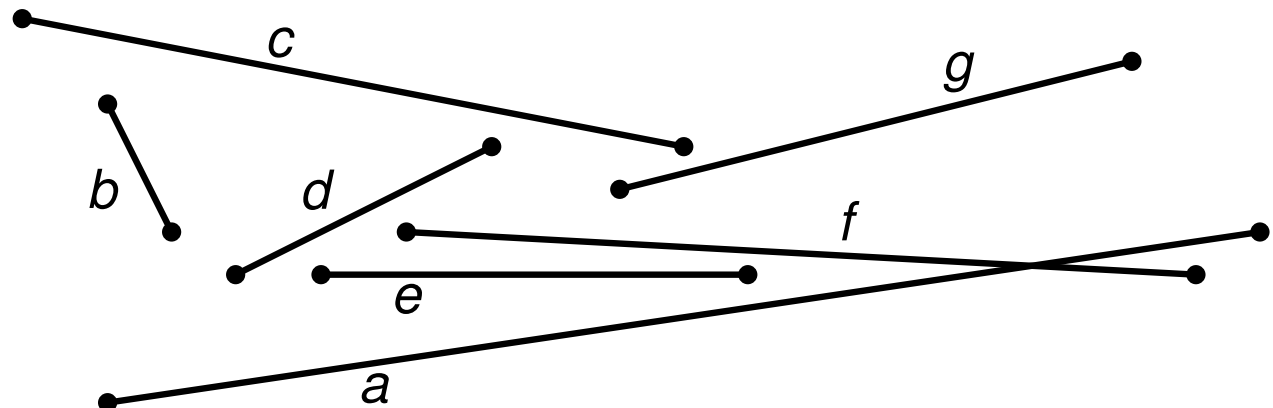
- Was bedeuten die Begriffe Strecke, (einfaches/konvexes) Polygon, konvexe Hülle, umschließendes Rechteck?
- Wie testet man, auf welcher Seite einer Gerade ein Punkt liegt?
- Wie testet man, ob zwei Strecken sich schneiden?

Algorithmisches

- Was ist die Grundidee eines Sweep-Line Algorithmus? Was speichert man im Laufe des Algorithmus?
- Wie und mit welcher Laufzeit kann man testen, ob es unter einer Menge von Strecken ein Paar gibt, das sich schneidet?

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand



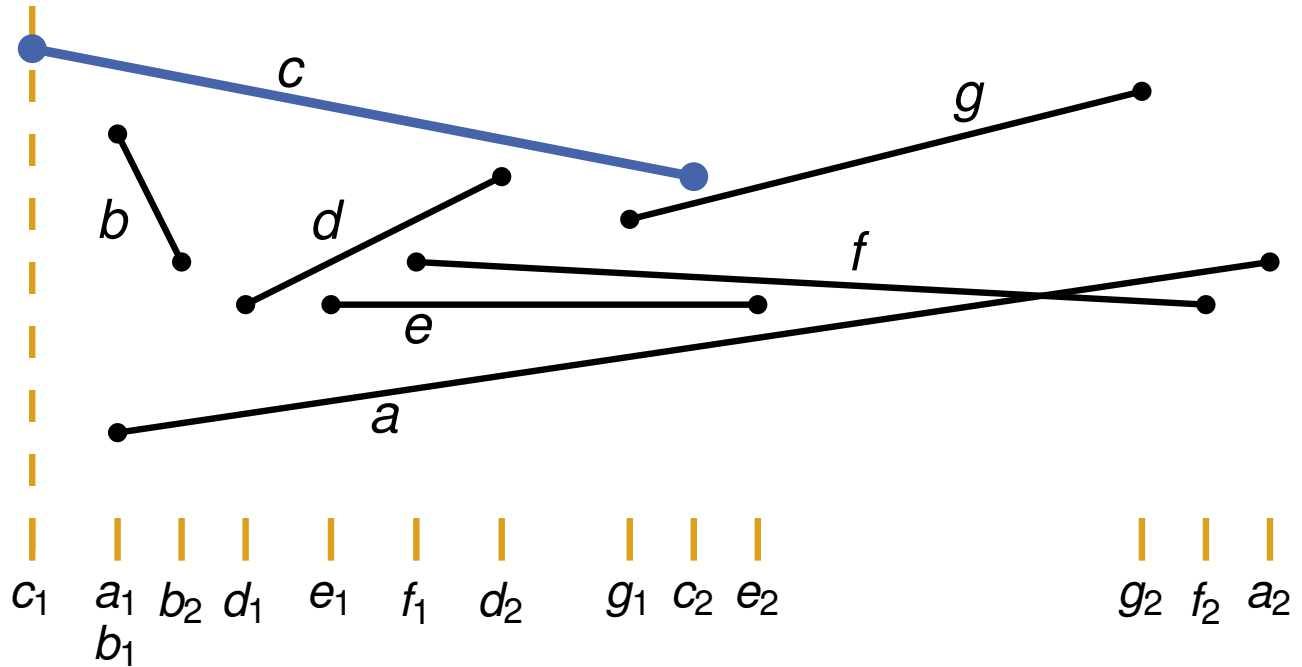
Event-Point Schedule →



Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

↓
+ c



Event-Point Schedule →

Sweep-Line Algorithmus – Beispiel

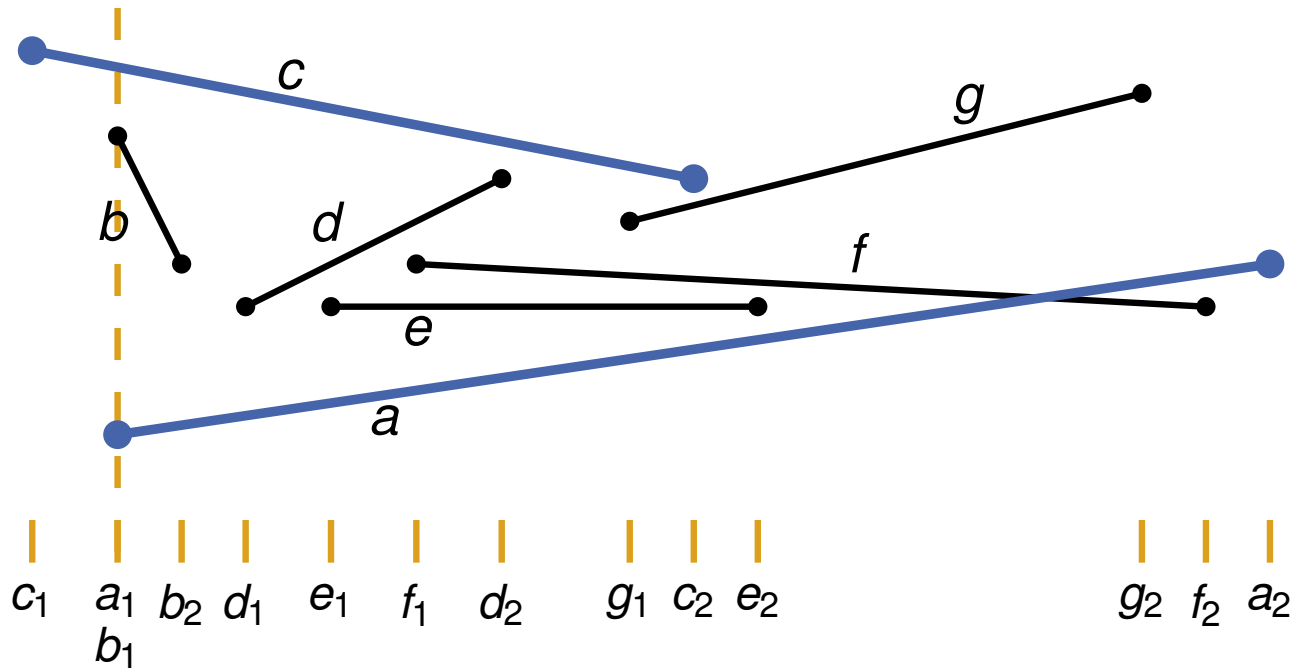
Sweep-Line Zustand



c

+ a

Event-Point Schedule →



a schneidet c nicht

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

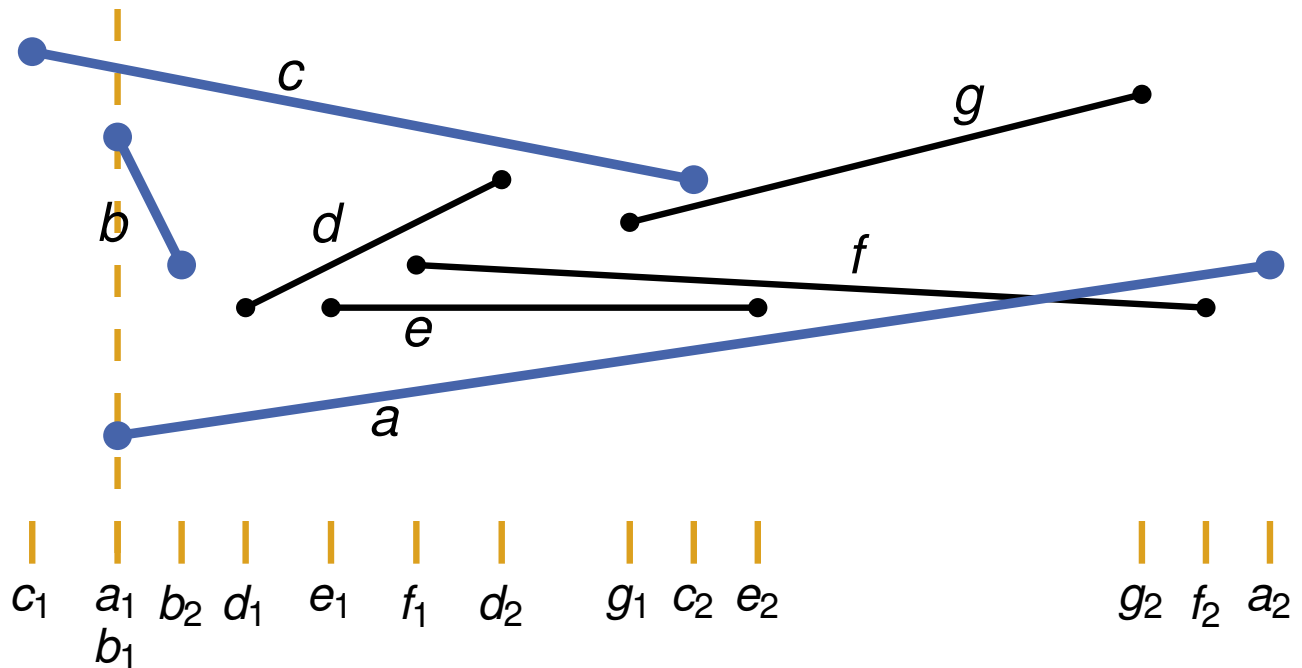


c

+ b

a

Event-Point Schedule →



b schneidet weder a noch c

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

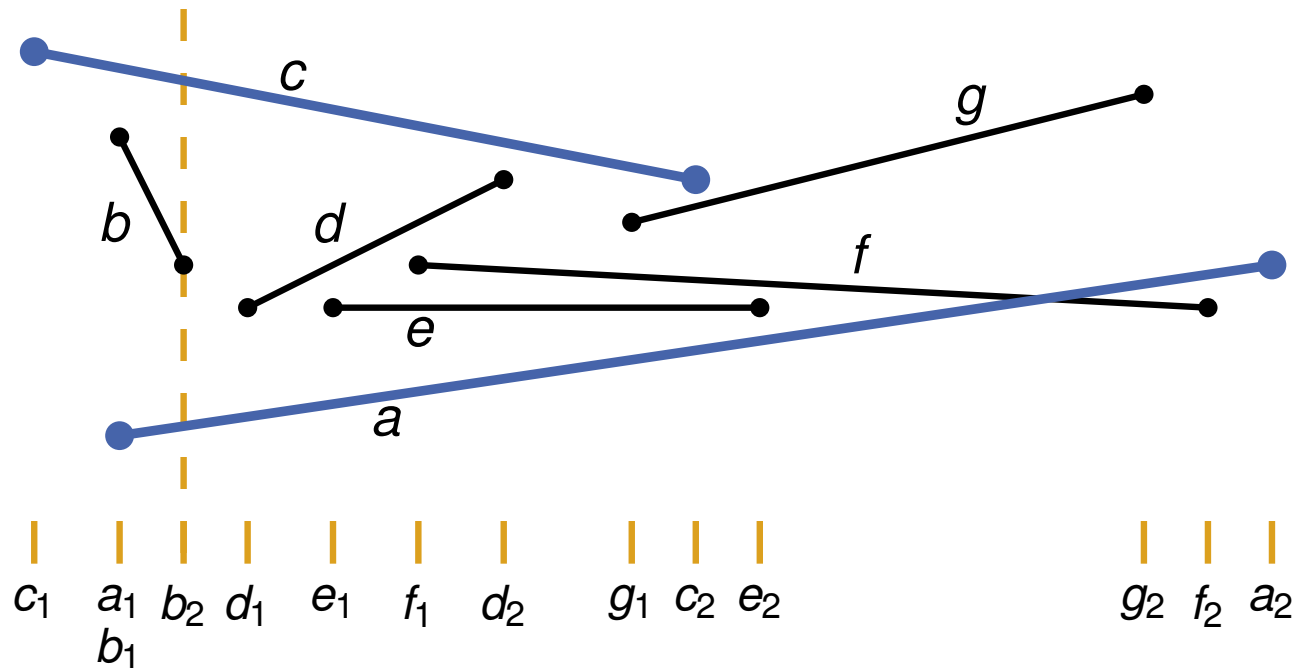


c

~~*b*~~

a

Event-Point Schedule →



a schneidet *c* nicht

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

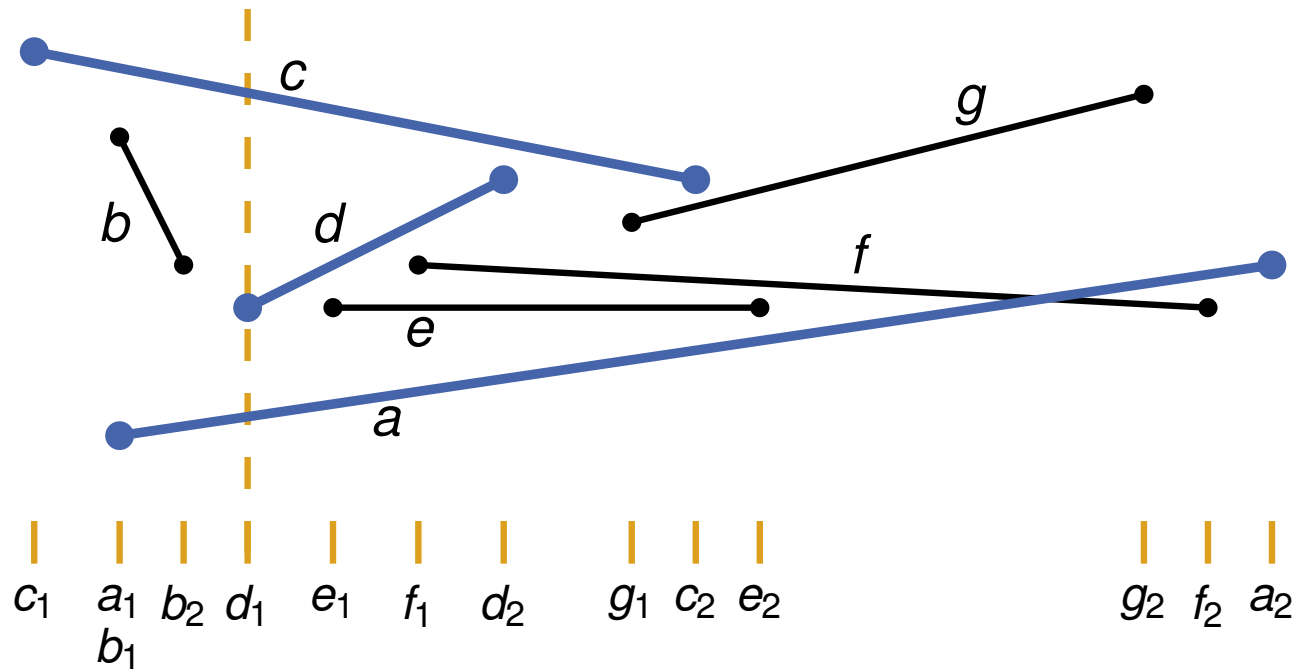


c

+ d

a

Event-Point Schedule →



d schneidet weder a noch c

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand



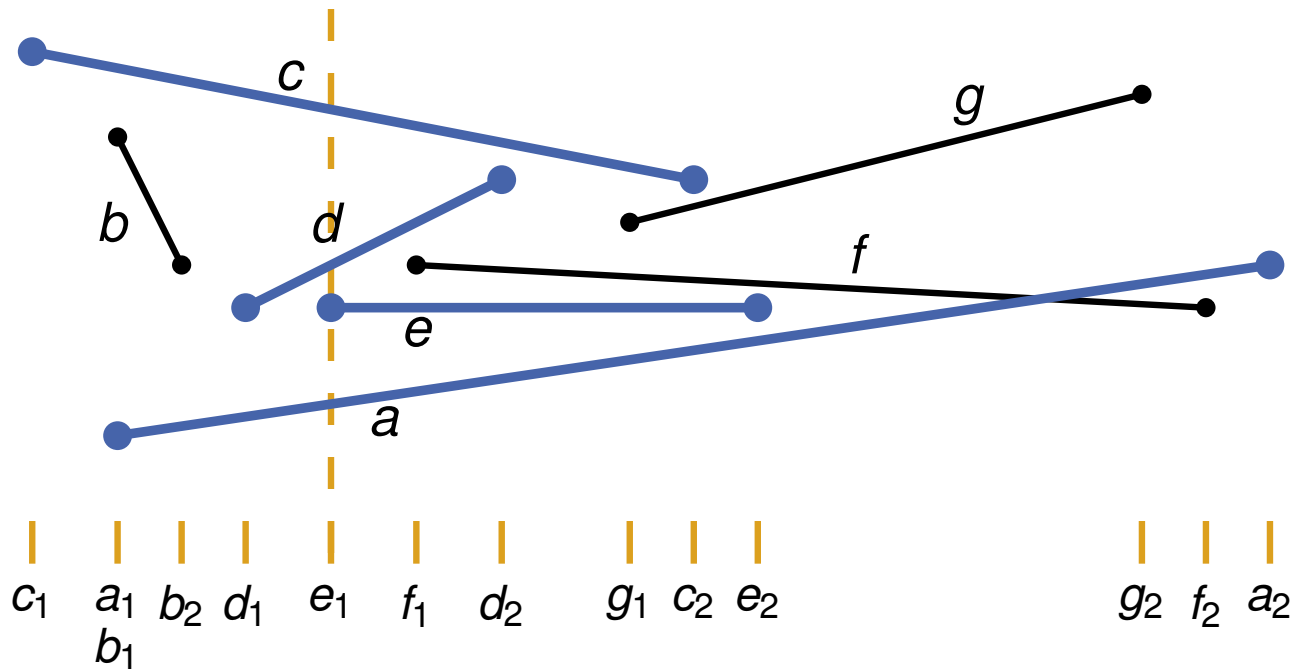
c

d

+ *e*

a

Event-Point Schedule →



e schneidet weder *d* noch *a*

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand



c

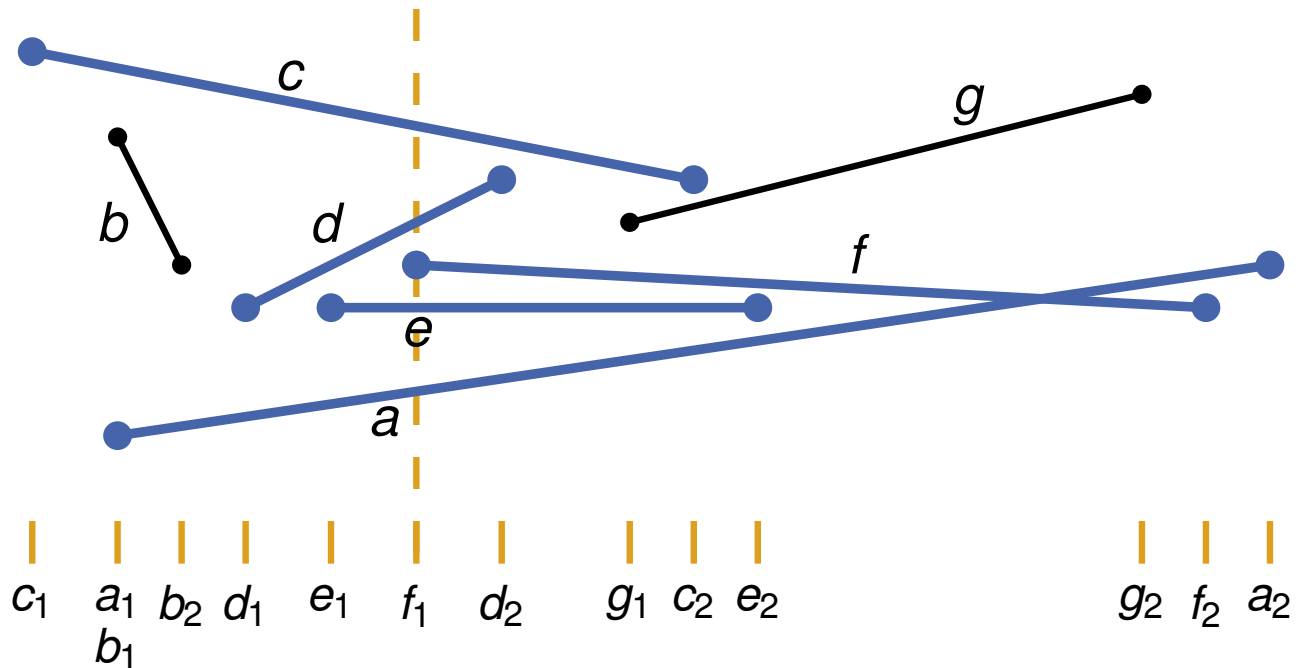
d

+ *f*

e

a

Event-Point Schedule →



f schneidet weder *d* noch *e*

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand



c

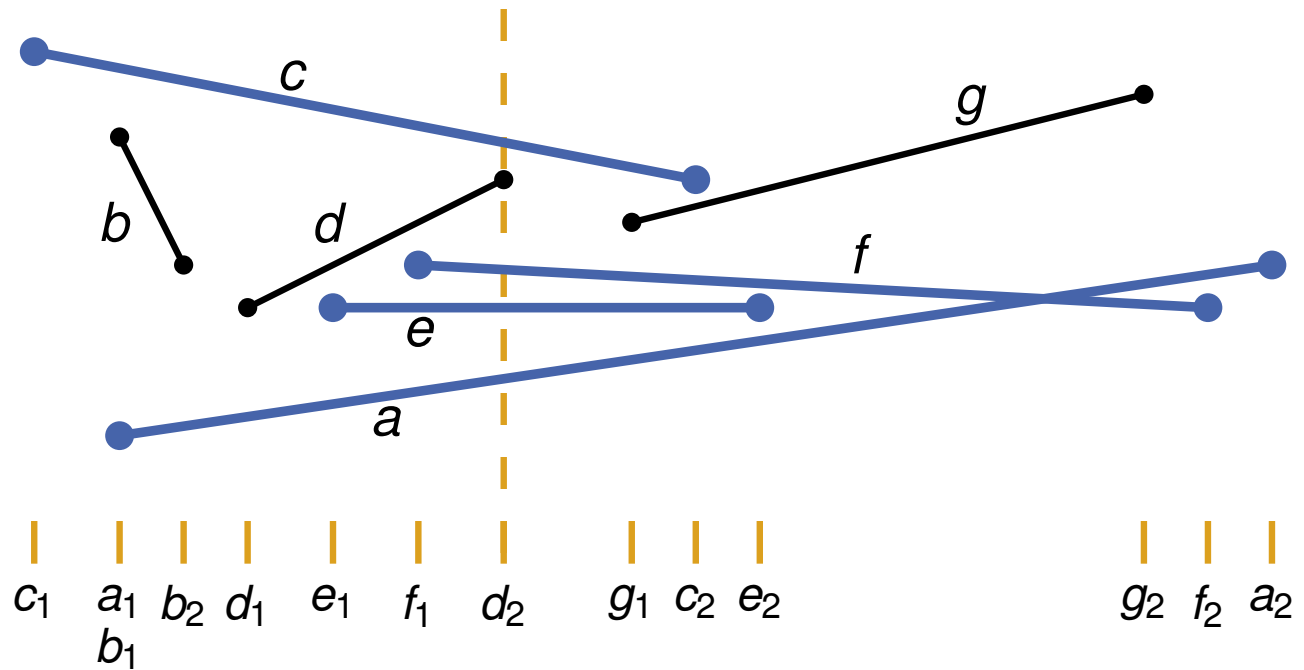
~~*d*~~

f

e

a

Event-Point Schedule →

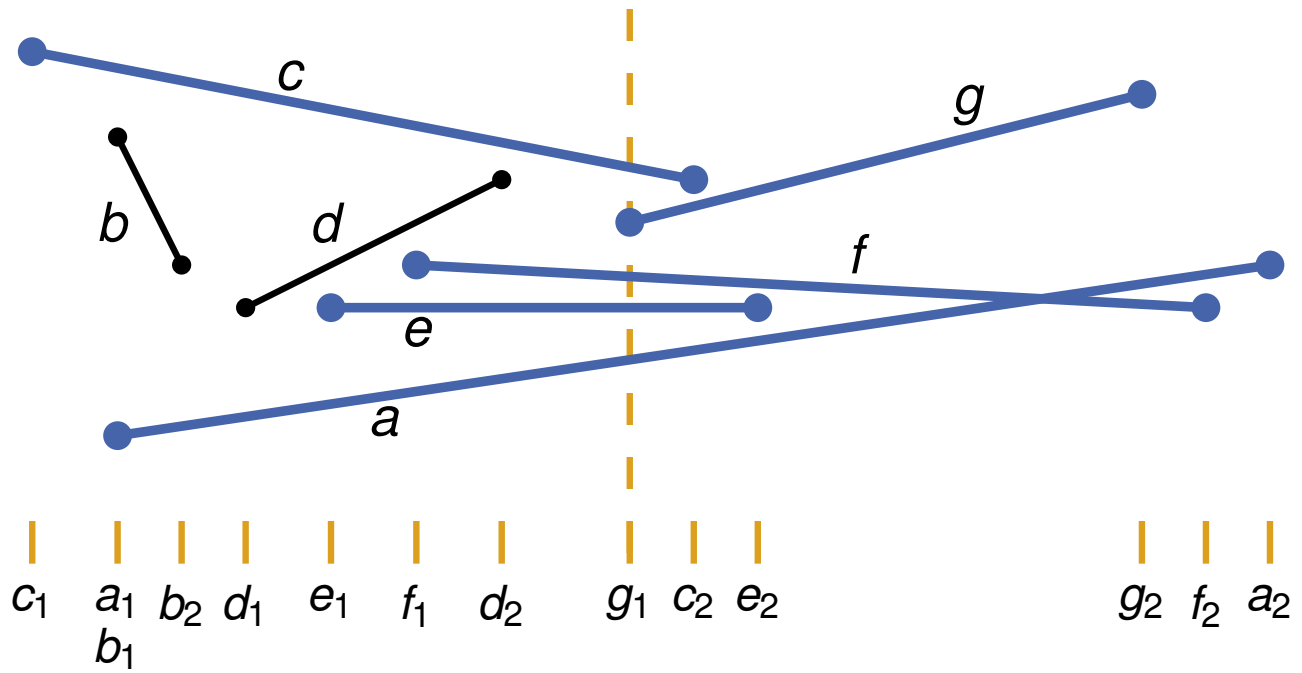


f schneidet *c* nicht

Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

↓
c
 + *g*
f
e
a



Event-Point Schedule →

g schneidet weder *f* noch *c*

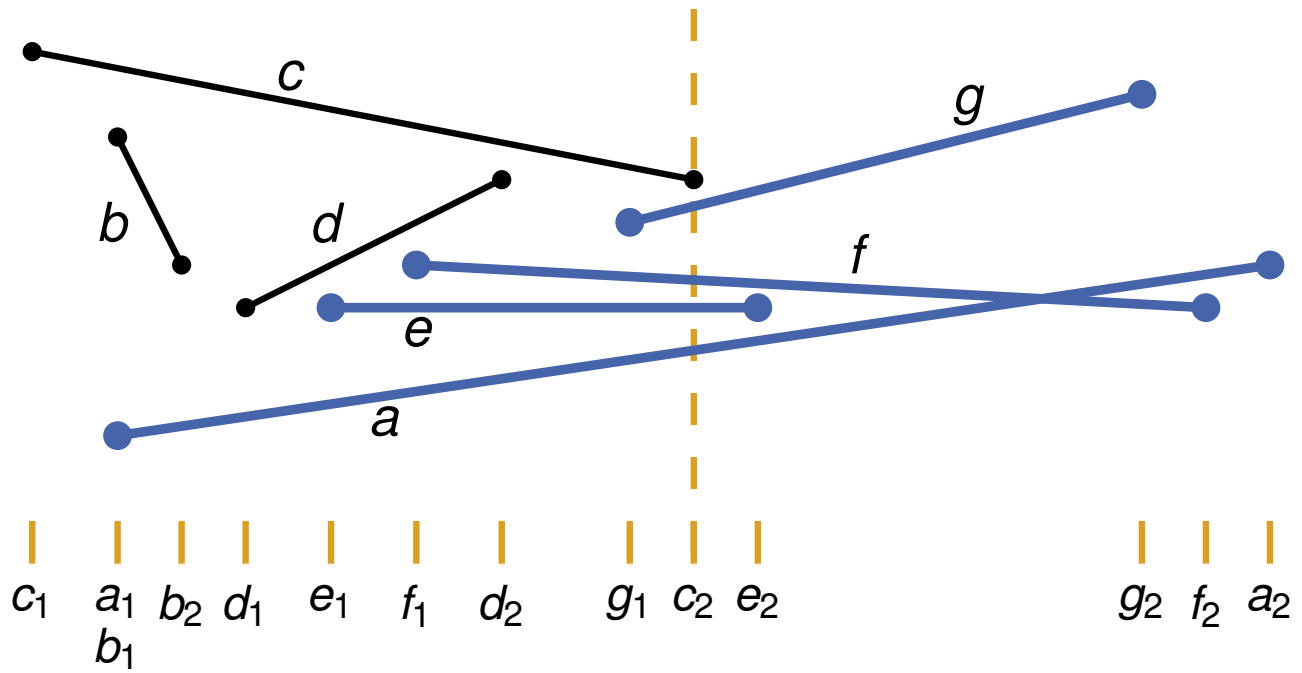
Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand



~~c~~
g
f
e
a

Event-Point Schedule →



Sweep-Line Algorithmus – Beispiel

Sweep-Line Zustand

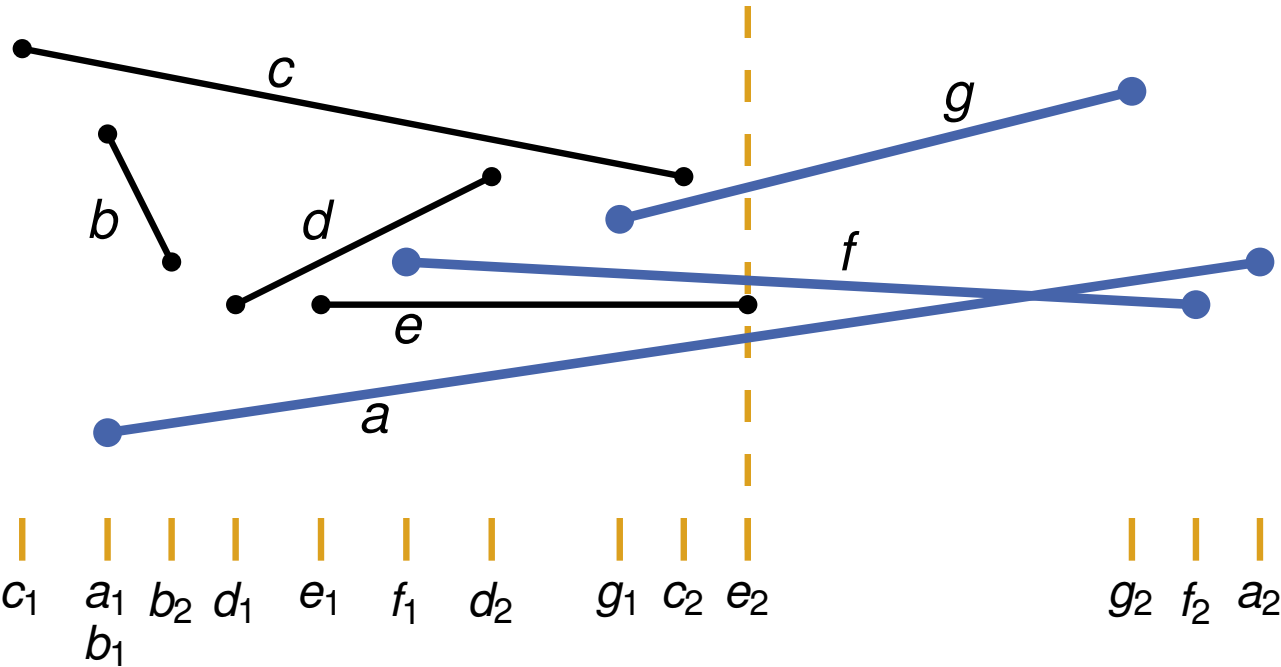


g

f

~~*e*~~

a



Event-Point Schedule →

f schneidet *a*! ⇒ gib TRUE zurück

Grundlegendes

- Was bedeuten die Begriffe Strecke, (einfaches/konvexes) Polygon, Konvexkombination konvex, konvexe Hülle, umschließendes Rechteck?
- Wie testet man, auf welcher Seite einer Gerade ein Punkt liegt?
- Wie testet man, ob zwei Strecken sich schneiden?

Algorithmisches

- Was ist die Grundidee eines Sweep-Line Algorithmus? Was speichert man im Laufe des Algorithmus?
- Wie und mit welcher Laufzeit kann man Testen, ob es unter einer Menge von Strecken ein Paar gibt, das sich schneidet?

Grundlegendes

- Was bedeuten die Begriffe Strecke, (einfaches/konvexes) Polygon, Konvexkombination konvex, konvexe Hülle, umschließendes Rechteck?
- Wie testet man, auf welcher Seite einer Gerade ein Punkt liegt?
- Wie testet man, ob zwei Strecken sich schneiden?

Algorithmisches

- Was ist die Grundidee eines Sweep-Line Algorithmus? Was speichert man im Laufe des Algorithmus?
- Wie und mit welcher Laufzeit kann man Testen, ob es unter einer Menge von Strecken ein Paar gibt, das sich schneidet?
- Was ist ein Quadtree? Für was werden Quadtrees gebraucht? Wie können sie effizient konstruiert werden?

Grundlegendes

- Was bedeuten die Begriffe Strecke, (einfaches/konvexes) Polygon, Konvexkombination konvex, konvexe Hülle, umschließendes Rechteck?
- Wie testet man, auf welcher Seite einer Gerade ein Punkt liegt?
- Wie testet man, ob zwei Strecken sich schneiden?

Algorithmisches

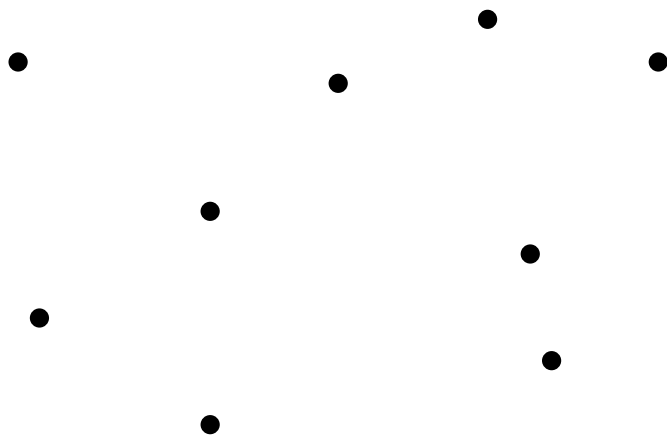
- Was ist die Grundidee eines Sweep-Line Algorithmus? Was speichert man im Laufe des Algorithmus?
- Wie und mit welcher Laufzeit kann man Testen, ob es unter einer Menge von Strecken ein Paar gibt, das sich schneidet?
- Was ist ein Quadtree? Für was werden Quadtrees gebraucht? Wie können sie effizient konstruiert werden?
- Wie funktionieren der Graham Scan bzw. der Jarvis' March (gift wrapping)? Welche Laufzeit haben sie?

Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:
Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



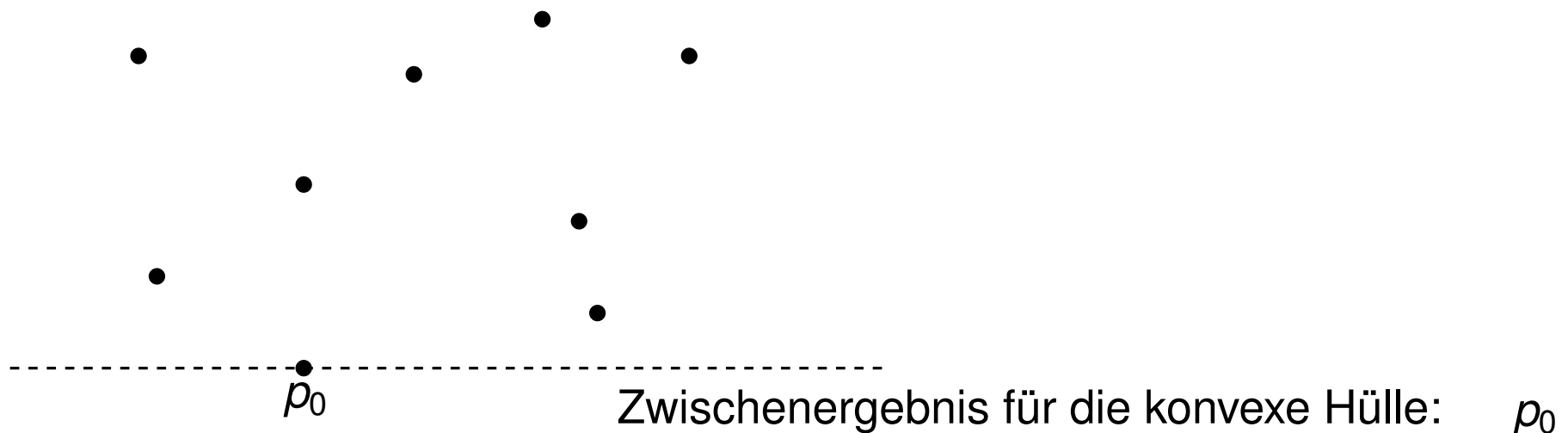
Zwischenergebnis für die konvexe Hülle:

Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:
Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:

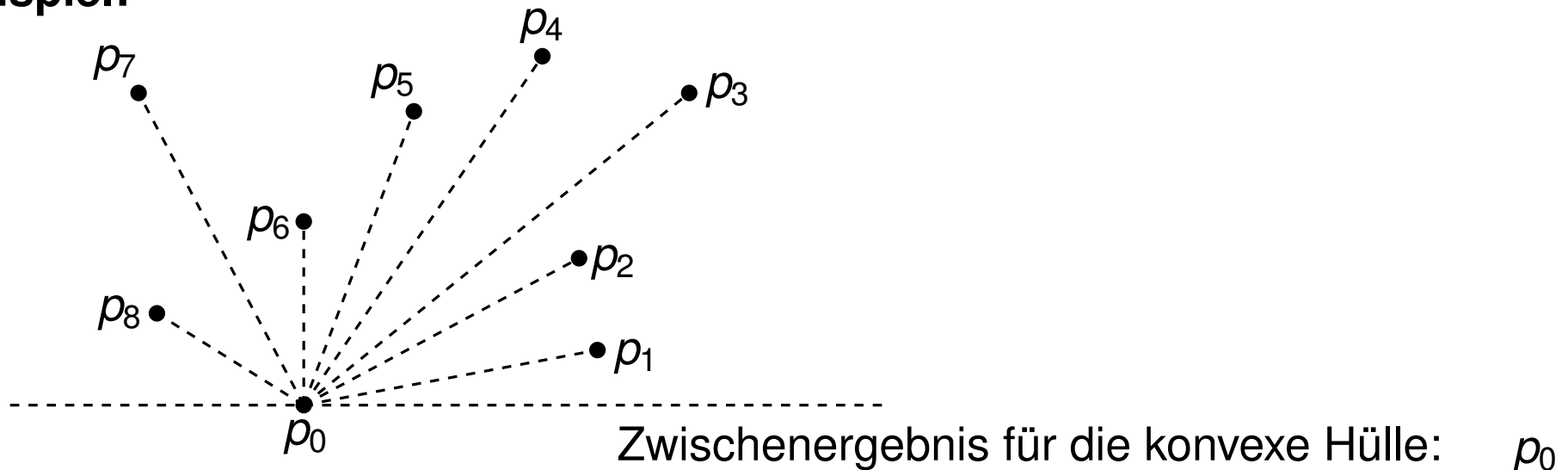


Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:
Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:

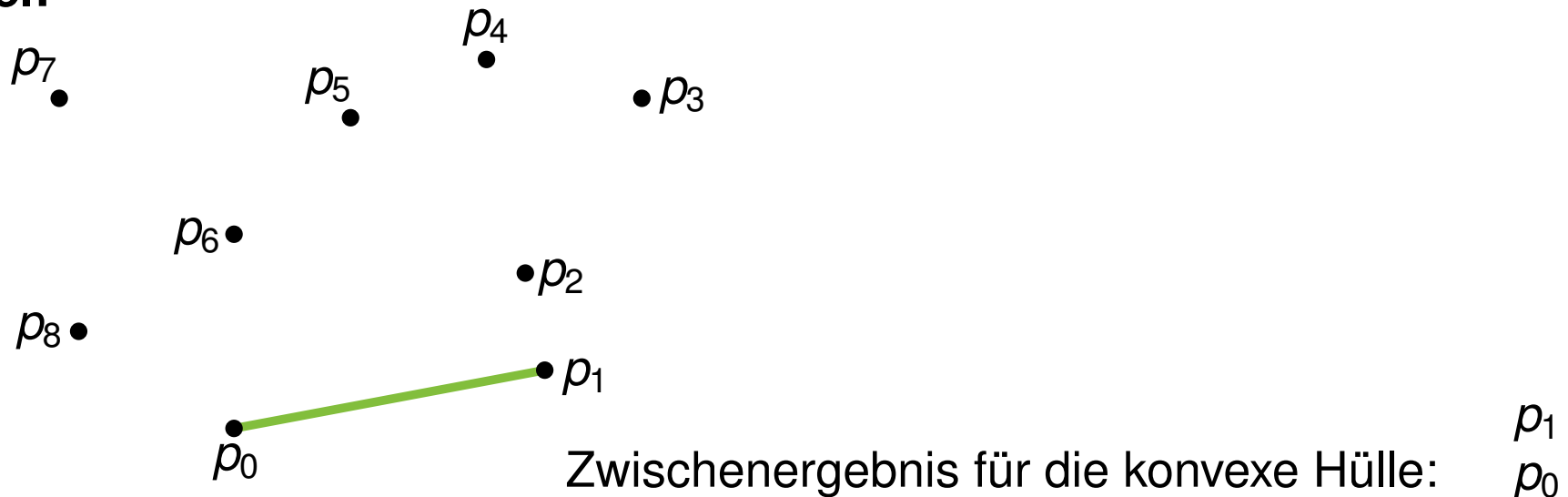


Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:
Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



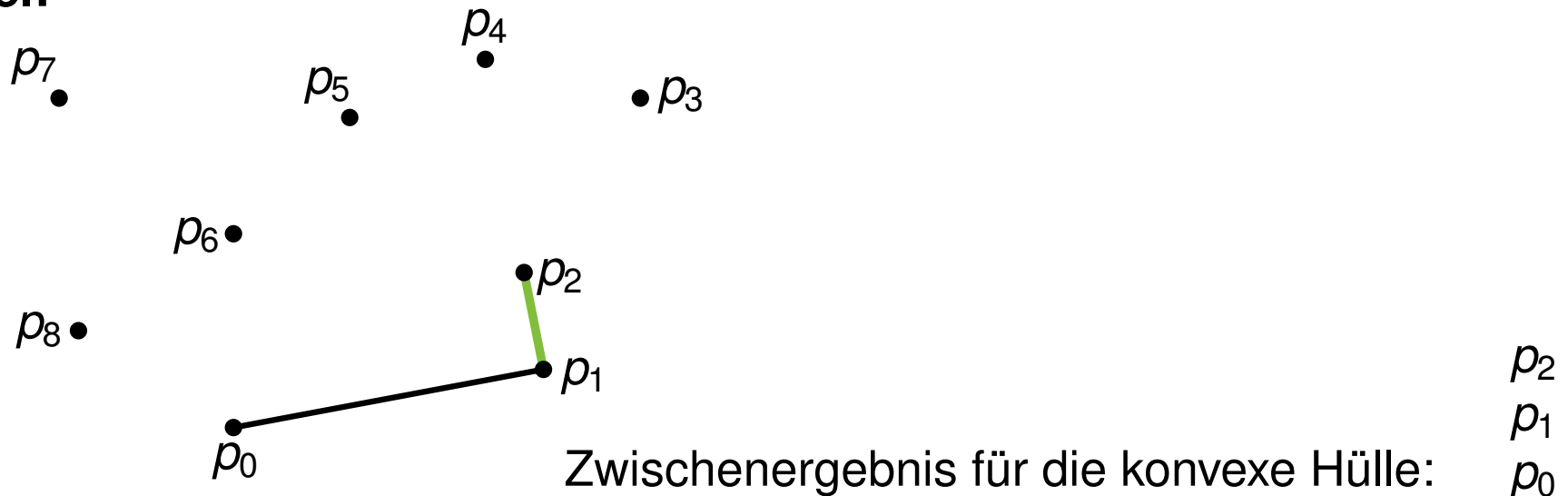
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



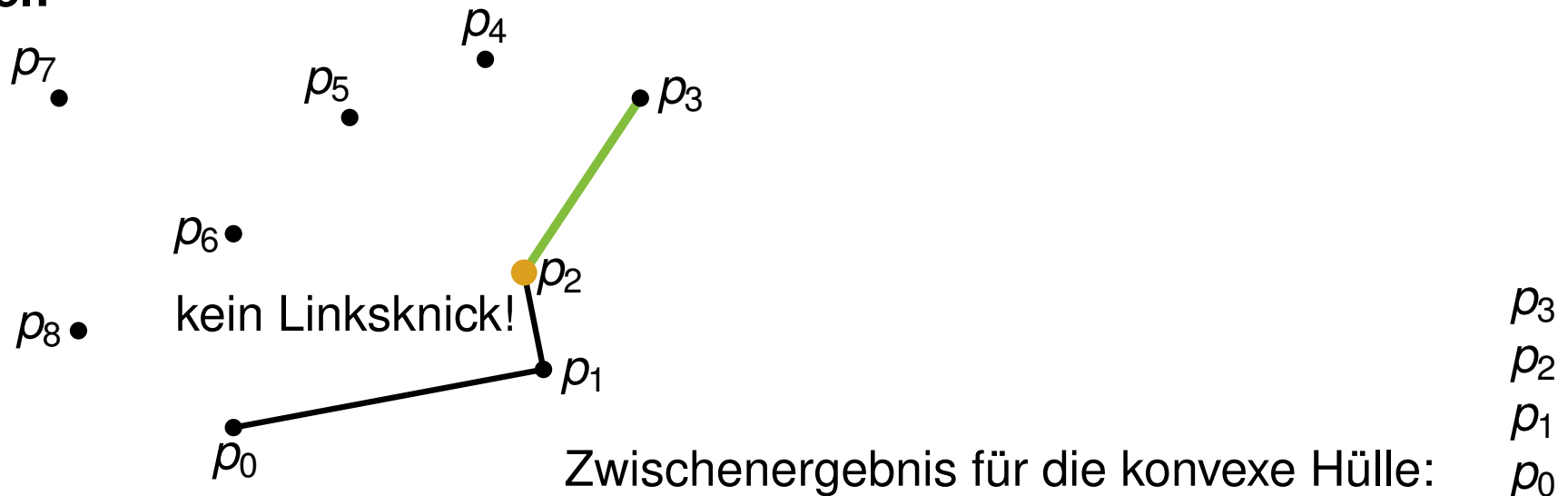
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:

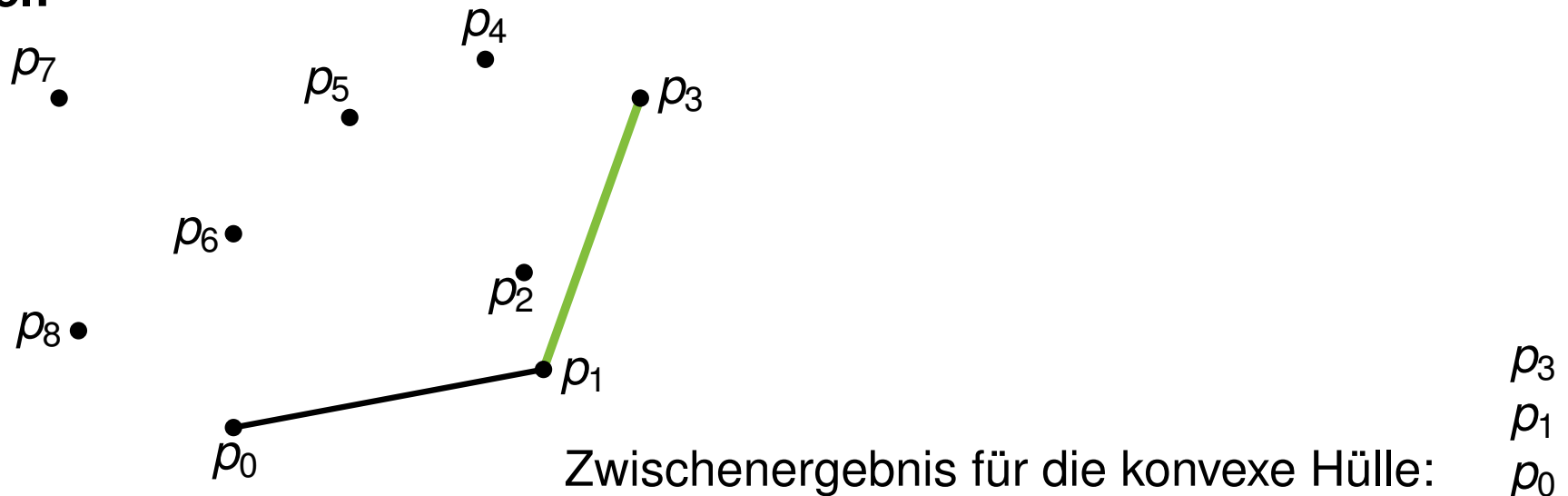


Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:
Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



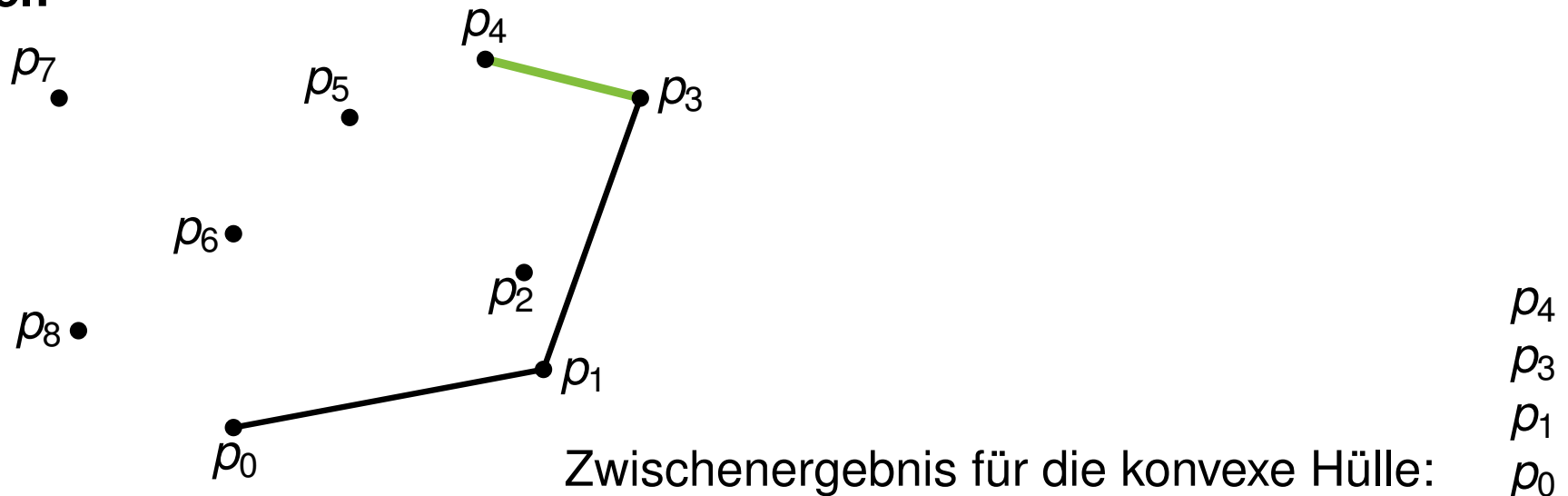
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



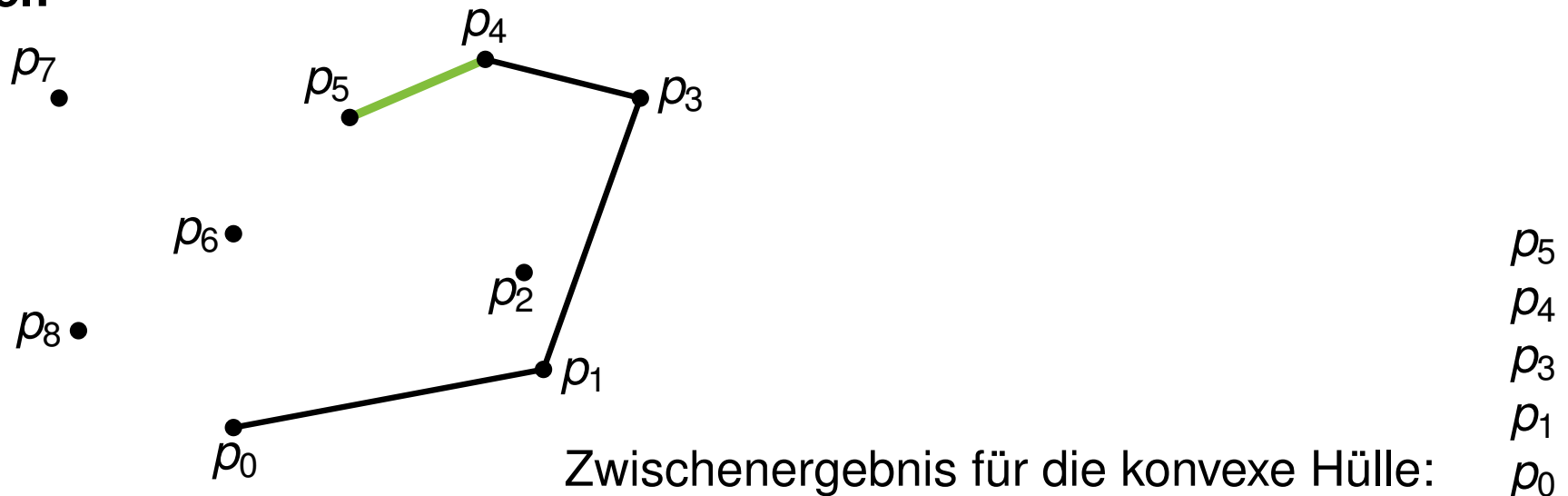
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



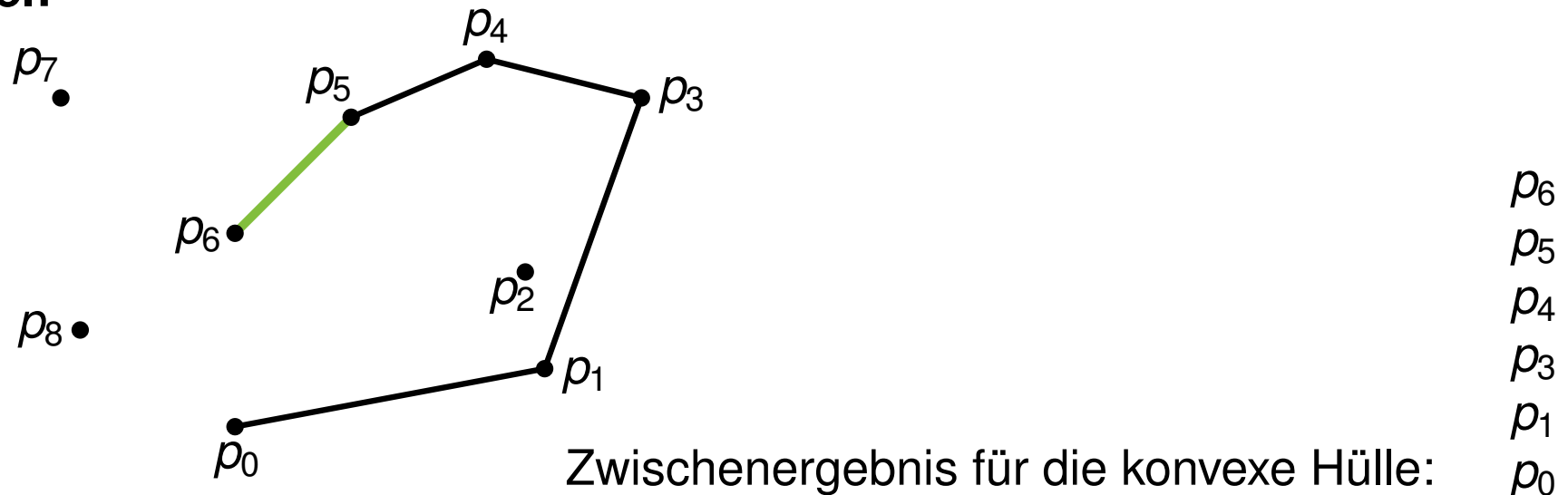
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



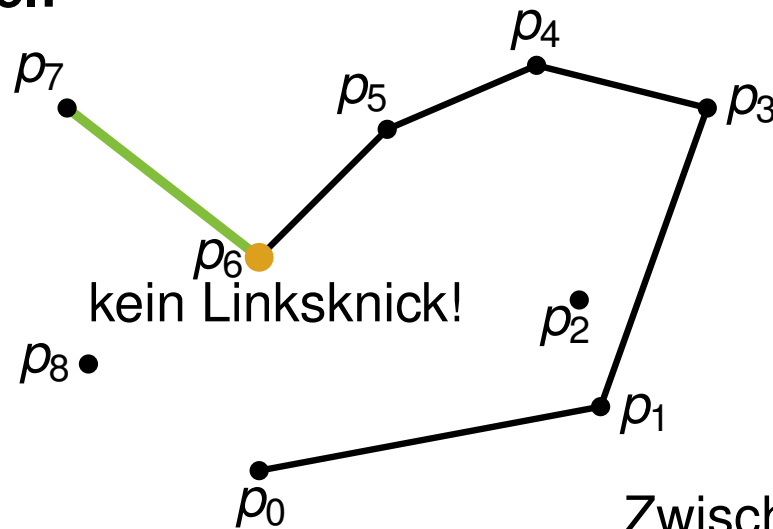
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_7
 p_6
 p_5
 p_4
 p_3
 p_1
 p_0

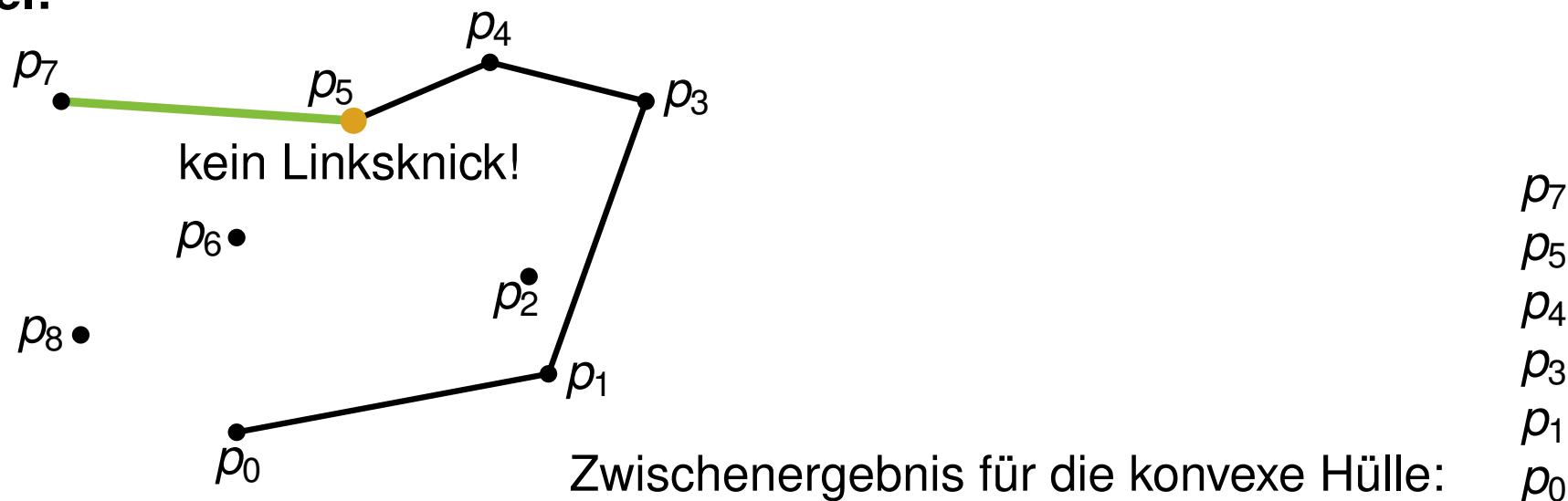
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



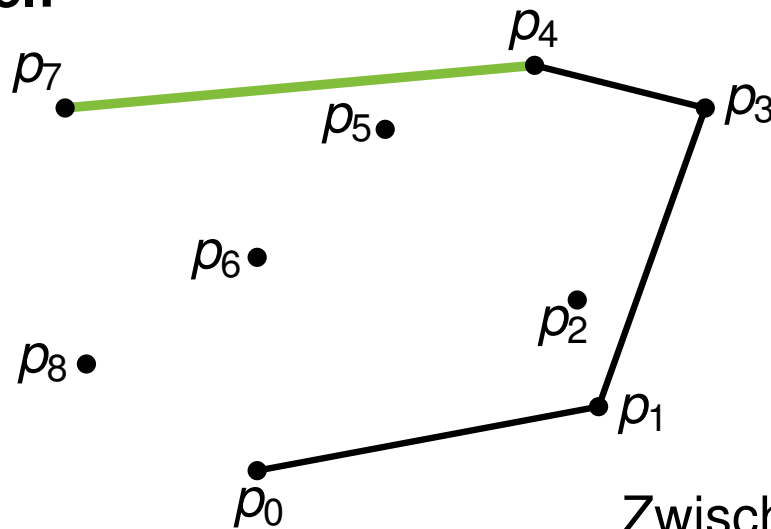
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_7
 p_4
 p_3
 p_1
 p_0

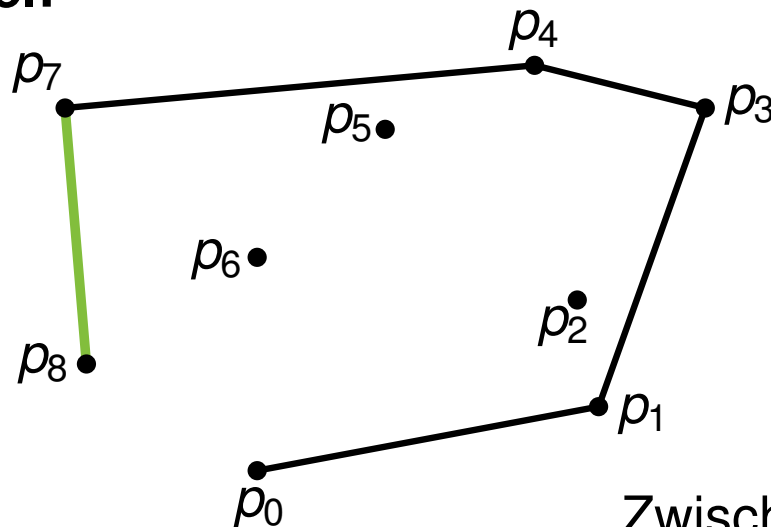
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_8
 p_7
 p_4
 p_3
 p_1
 p_0

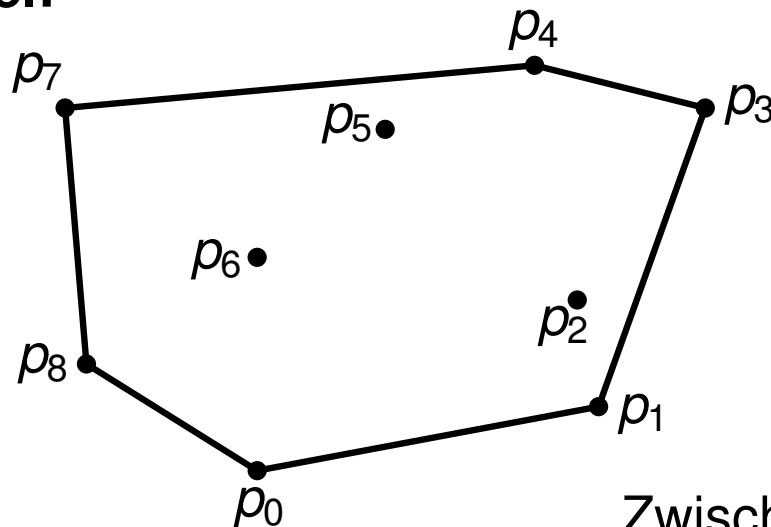
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linksten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte p_0, p_1, \dots, p_{n-1} in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_8
 p_7
 p_4
 p_3
 p_1
 p_0

String-Matching

Grundlegendes

- Was ist die Problemstellung?
- Warum ist $\Omega(n + m)$ eine untere Schranke?
- Wie hängt die Wahl des Algorithmus von der Art der Anfrage ab?

String-Matching

Grundlegendes

- Was ist die Problemstellung?
- Warum ist $\Omega(n + m)$ eine untere Schranke?
- Wie hängt die Wahl des Algorithmus von der Art der Anfrage ab?

Algorithmisches

- Was ist die Idee bei Rabin & Karp?

Grundlegendes

- Was ist die Problemstellung?
- Warum ist $\Omega(n + m)$ eine untere Schranke?
- Wie hängt die Wahl des Algorithmus von der Art der Anfrage ab?

Algorithmisches

- Was ist die Idee bei Rabin & Karp?
- Wie funktioniert String-Matching mit endlichen Automaten? Laufzeit?

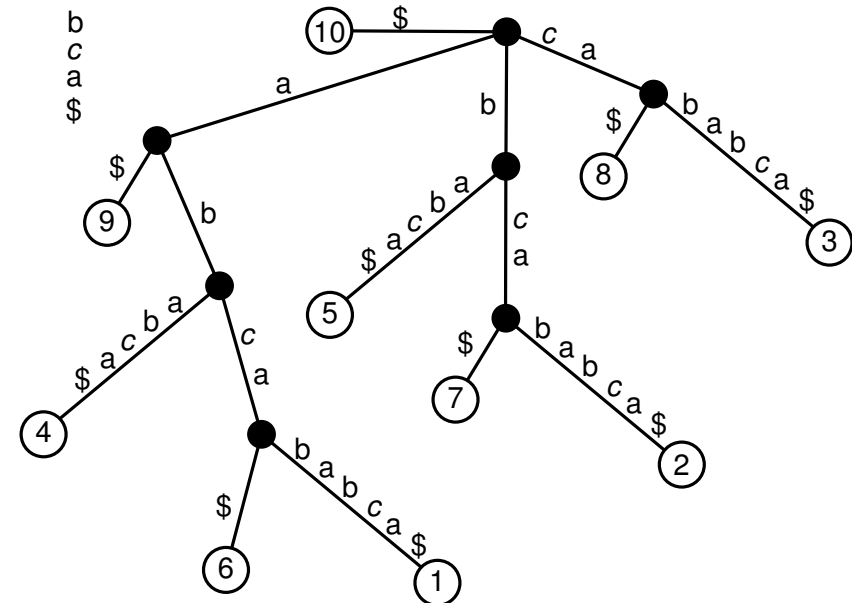
String-Matching

Beispiel: $T = a b c a b a b c a \$$

A =	10	9	4	6	1	5	7	2	8	3
\$	a	a	a	a	b	b	b	c	c	
\$	b	b	b	b	a	c	c	a	a	
	a	c	c	b	a	a	\$	b		
	b	a	a	c	\$	b	a	b		
	c	\$	b	a		a	b	c		
	a		a	a		a	b	c		
	\$		b	c		a				

Grundlegendes

- Was ist die Problemstellung?
- Warum ist $\Omega(n + m)$ eine untere Schranke?
- Wie hängt die Wahl des Algorithmus von der Art der Anfrage ab?



Algorithmisches

- Was ist die Idee bei Rabin & Karp?
- Wie funktioniert String-Matching mit endlichen Automaten? Laufzeit?
- Was ist ein Suffix-Baum? Was ist ein Suffix-Array?
- Wie kann man mit diesen Datenstrukturen Muster finden? Laufzeit?

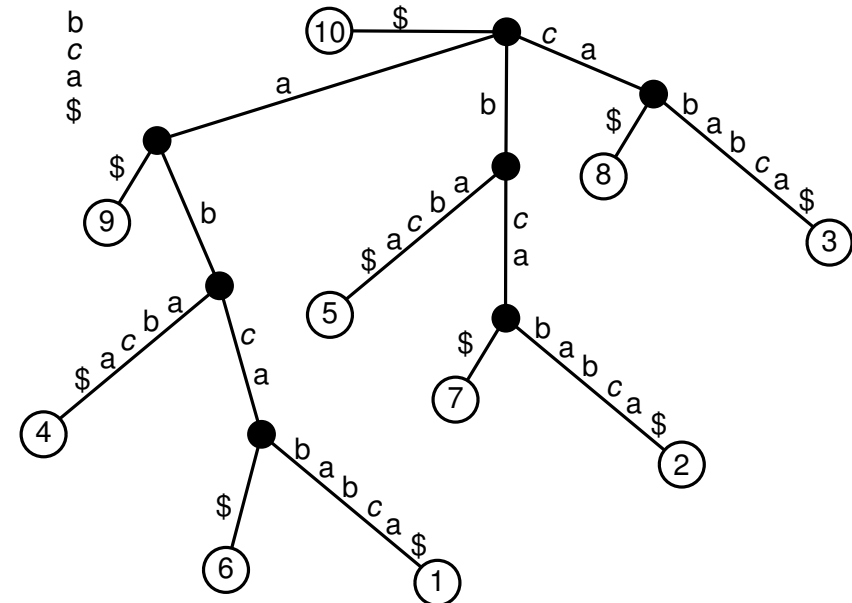
String-Matching

Beispiel: $T = abcababca\$$

A =	10	9	4	6	1	5	7	2	8	3
\$	a	a	a	a	b	b	b	c	c	
\$	b	b	b	a	c	c	a	a		
	a	c	c	b	a	a	\$	b		
	b	a	a	c	\$	b	a	b		
	c	\$	b	a		a	b	c		
	a		a	a		a	b	c		
	\$		b	c		a	a	\$		

Grundlegendes

- Was ist die Problemstellung?
- Warum ist $\Omega(n + m)$ eine untere Schranke?
- Wie hängt die Wahl des Algorithmus von der Art der Anfrage ab?



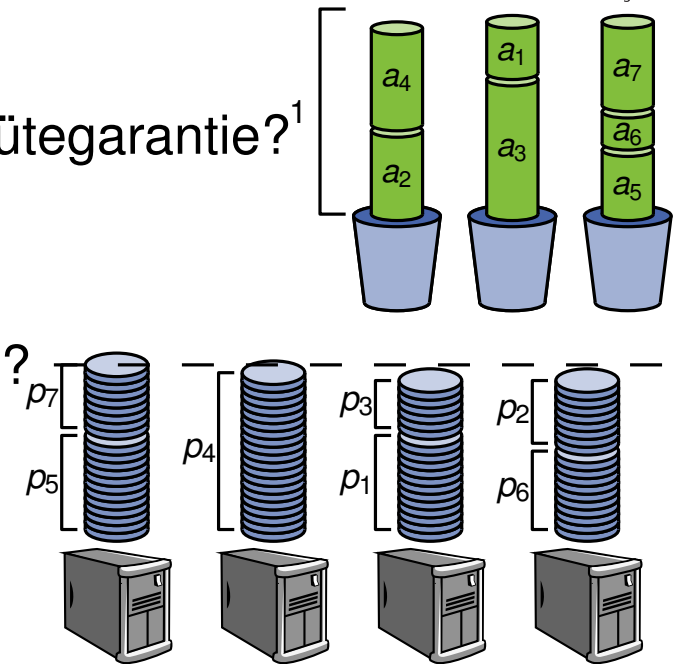
Algorithmisches

- Was ist die Idee bei Rabin & Karp?
- Wie funktioniert String-Matching mit endlichen Automaten? Laufzeit?
- Was ist ein Suffix-Baum? Was ist ein Suffix-Array?
- Wie kann man mit diesen Datenstrukturen Muster finden? Laufzeit?
- In welcher Laufzeit können Suffix-Bäume und Suffix-Arrays konstruiert werden?
- Wie kann man sie konstruieren?

Approximierende Algorithmen

Grundlegendes

- Was sind Approximationsalgorithmen mit relativer Gütegarantie?¹
- Was sind PAS, FPAS, APAS, AFPAS?
- Wie sind Travelling Salesman Problem, Knapsack, Multiprozessor-Scheduling und Bin Packing definiert?



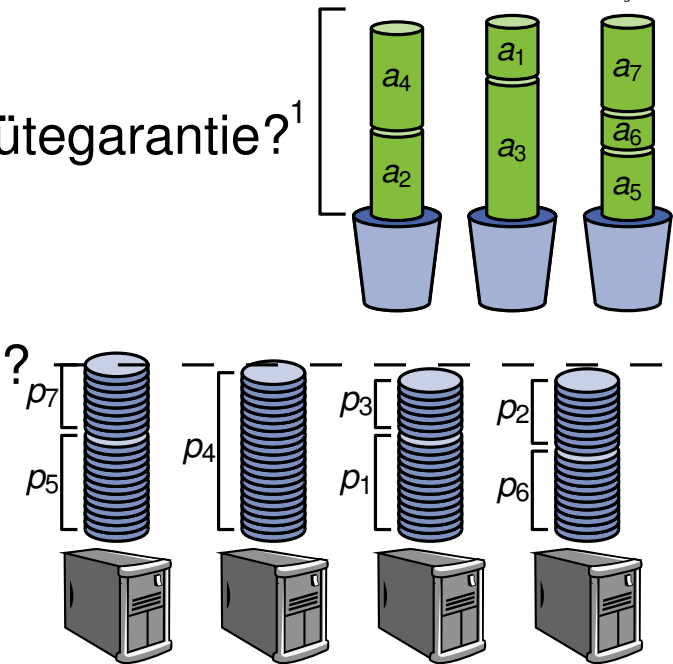
Approximierende Algorithmen

Grundlegendes

- Was sind Approximationsalgorithmen mit relativer Gütegarantie?¹
- Was sind PAS, FPAS, APAS, AFPAS?
- Wie sind Travelling Salesman Problem, Knapsack, Multiprozessor-Scheduling und Bin Packing definiert?

Algorithmisches

- 2-Approximation für Travelling Salesman
- Wie funktioniert dyn. Programm für Knapsack?
- Wie funktioniert FPAS für Knapsack?



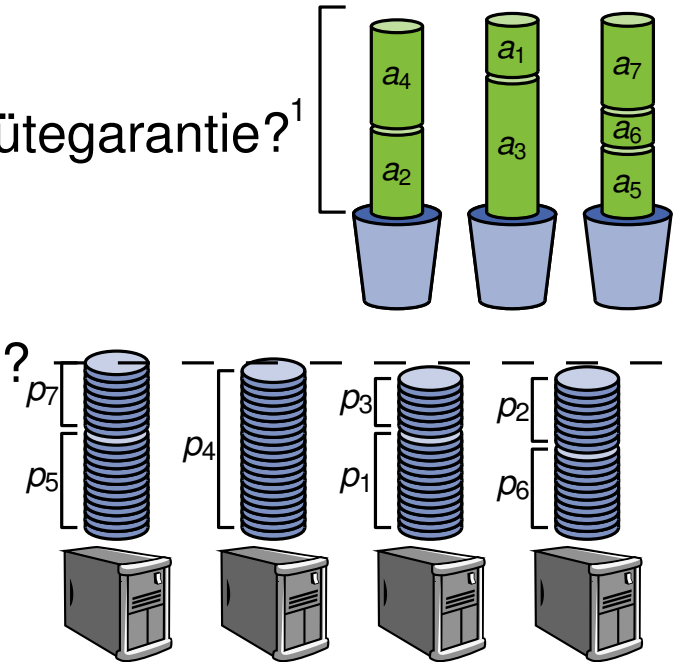
Approximierende Algorithmen

Grundlegendes

- Was sind Approximationsalgorithmen mit relativer Gütegarantie?¹
- Was sind PAS, FPAS, APAS, AFPAS?
- Wie sind Travelling Salesman Problem, Knapsack, Multiprozessor-Scheduling und Bin Packing definiert?

Algorithmisches

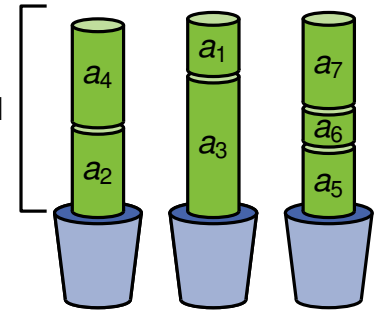
- 2-Approximation für Travelling Salesman
- Wie funktioniert dyn. Programm für Knapsack?
- Wie funktioniert FPAS für Knapsack?
- Wie funktioniert LIST SCHEDULING und was ist die Approximationsgüte? Wie kann man damit zu einem PAS kommen?



Approximierende Algorithmen

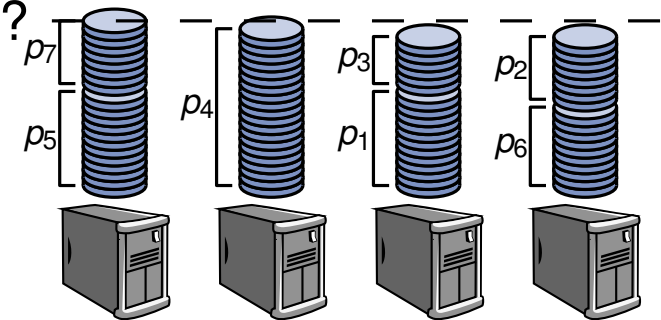
Grundlegendes

- Was sind Approximationsalgorithmen mit relativer Gütegarantie?¹
- Was sind PAS, FPAS, APAS, AFPAS?
- Wie sind Travelling Salesman Problem, Knapsack, Multiprozessor-Scheduling und Bin Packing definiert?



Algorithmisches

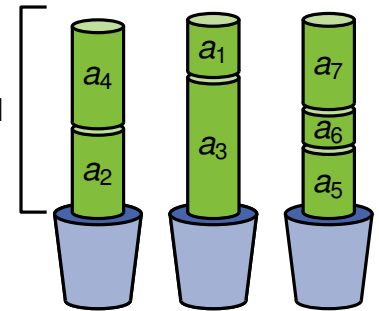
- 2-Approximation für Travelling Salesman
- Wie funktioniert dyn. Programm für Knapsack?
- Wie funktioniert FPAS für Knapsack?
- Wie funktioniert LIST SCHEDULING und was ist die Approximationsgüte? Wie kann man damit zu einem PAS kommen?
- Was ist NEXT FIT (Bin Packing)? Ist die Approximationsrate von 2 scharf?
- Was ist FIRST FIT? Wie ist die (asymptotische) Approximationsrate hier?



Approximierende Algorithmen

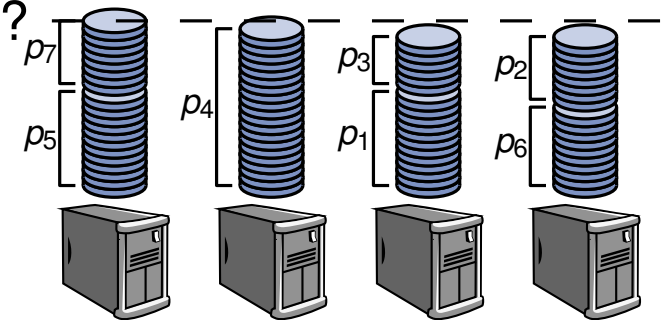
Grundlegendes

- Was sind Approximationsalgorithmen mit relativer Gütegarantie?¹
- Was sind PAS, FPAS, APAS, AFPAS?
- Wie sind Travelling Salesman Problem, Knapsack, Multiprozessor-Scheduling und Bin Packing definiert?



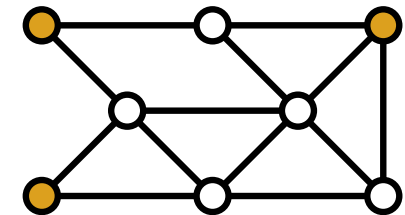
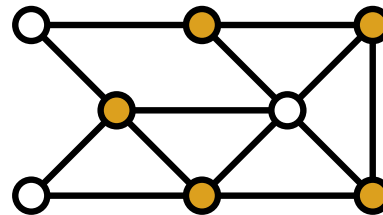
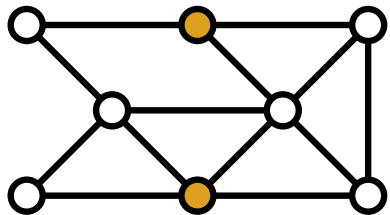
Algorithmisches

- 2-Approximation für Travelling Salesman
- Wie funktioniert dyn. Programm für Knapsack?
- Wie funktioniert FPAS für Knapsack?
- Wie funktioniert LIST SCHEDULING und was ist die Approximationsgüte? Wie kann man damit zu einem PAS kommen?
- Was ist NEXT FIT (Bin Packing)? Ist die Approximationsrate von 2 scharf?
- Was ist FIRST FIT? Wie ist die (asymptotische) Approximationsrate hier?
- Was ist RESTRICTED BIN PACKING (wenige verschiedene Elementtypen + Mindestgröße) und wie kann es als ILP formuliert werden?
- Wie funktioniert das APAS (AFPAS)?



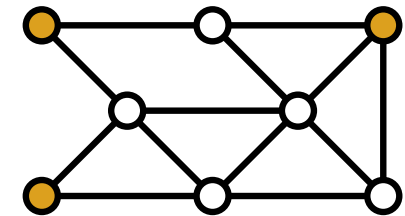
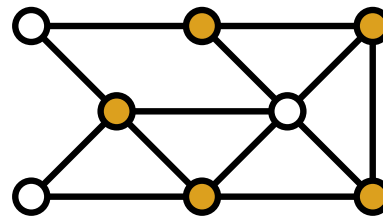
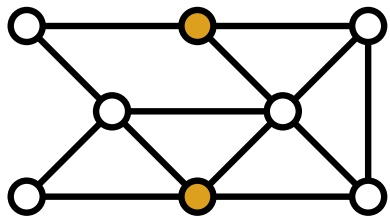
Grundlegendes

- Wie sind die Probleme VERTEX COVER, INDEPENDENT SET und DOMINATING SET definiert?
- Was ist ein FPT-Algorithmus?



Grundlegendes

- Wie sind die Probleme VERTEX COVER, INDEPENDENT SET und DOMINATING SET definiert?
- Was ist ein FPT-Algorithmus?

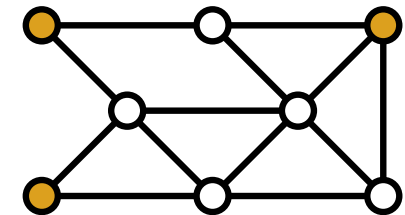
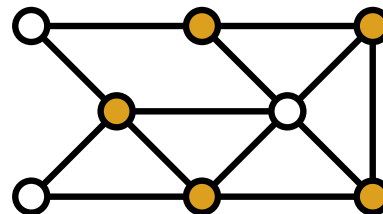
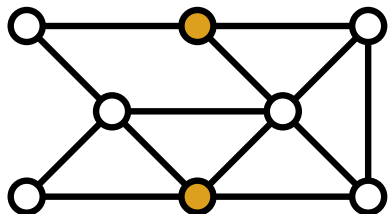


Algorithmisches

- Was ist Kernbildung? Wie funktionieren tiefenbeschränkte Suchbäume?
- Wie funktioniert die Kernbildung bei VERTEX COVER? Laufzeit?

Grundlegendes

- Wie sind die Probleme VERTEX COVER, INDEPENDENT SET und DOMINATING SET definiert?
- Was ist ein FPT-Algorithmus?

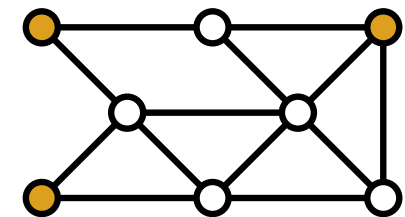
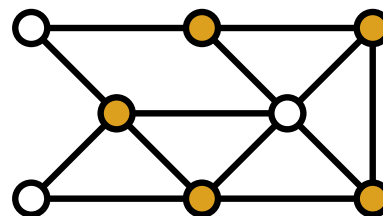
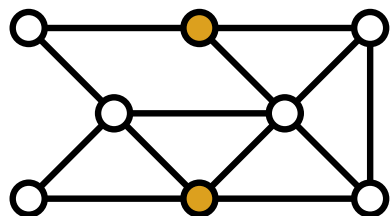


Algorithmisches

- Was ist Kernbildung? Wie funktionieren tiefenbeschränkte Suchbäume?
- Wie funktioniert die Kernbildung bei VERTEX COVER? Laufzeit?
- Wie funktioniert der Entscheidungsbaum für VERTEX COVER mit Verzweigungsvektor (1, 1)? Laufzeit?
- Idee, wie die Laufzeit verbessert werden kann?

Grundlegendes

- Wie sind die Probleme VERTEX COVER, INDEPENDENT SET und DOMINATING SET definiert?
- Was ist ein FPT-Algorithmus?



Algorithmisches

- Was ist Kernbildung? Wie funktionieren tiefenbeschränkte Suchbäume?
- Wie funktioniert die Kernbildung bei VERTEX COVER? Laufzeit?
- Wie funktioniert der Entscheidungsbaum für VERTEX COVER mit Verzweigungsvektor (1, 1)? Laufzeit?
- Idee, wie die Laufzeit verbessert werden kann?
- Wie kann VERTEX COVER als ILP formuliert werden? Wie hilft dieses ILP, wenn man einen FPT-Algorithmus sucht?

Grundlegendes

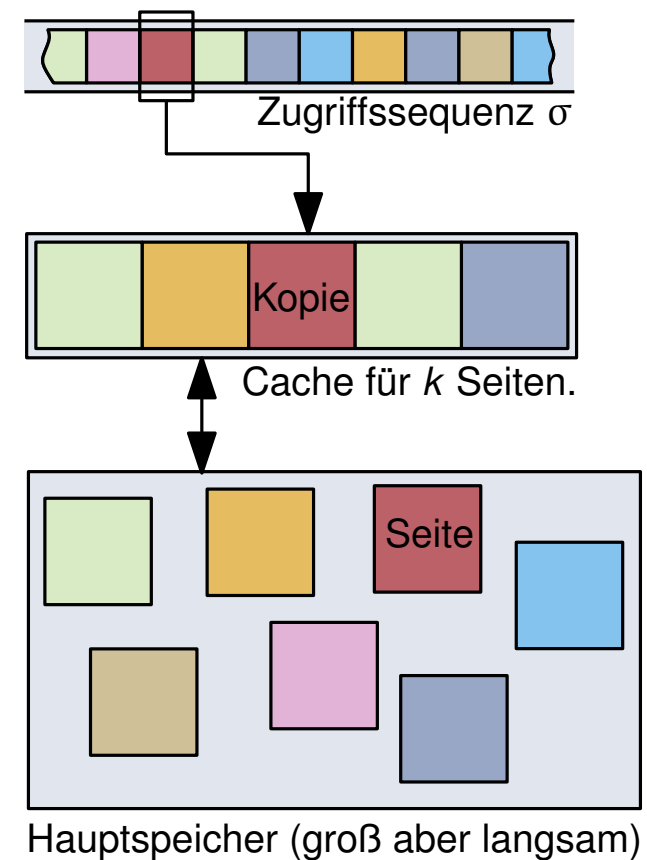
- Was ist ein Online-Algorithmus? Wann ist ein Algorithmus kompetitiv?
- Was hat das mit Approximationsalgorithmen zu tun?

Grundlegendes

- Was ist ein Online-Algorithmus? Wann ist ein Algorithmus kompetitiv?
- Was hat das mit Approximationsalgorithmen zu tun?

Paging

- Was ist die Problemstellung? Welche Strategien gibt es?

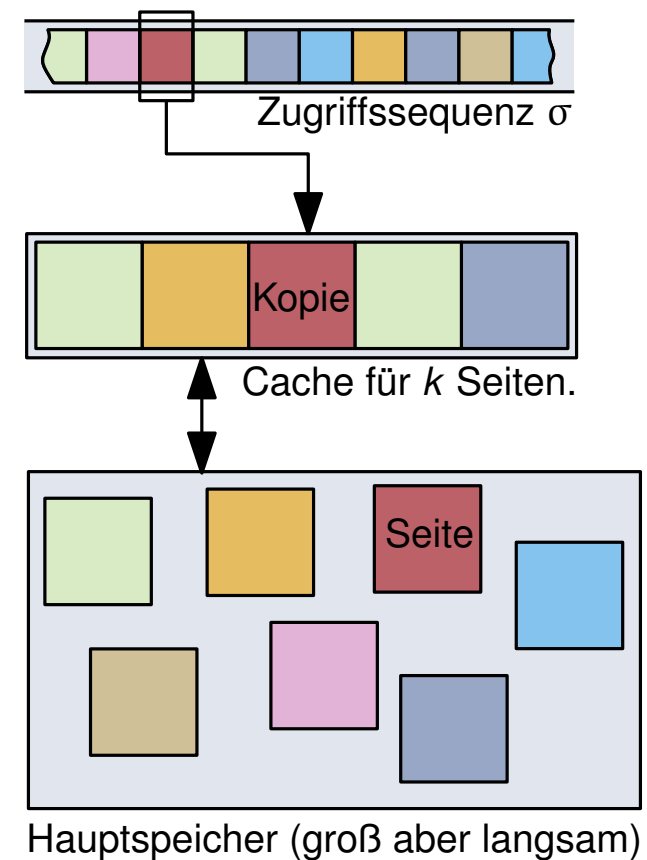


Grundlegendes

- Was ist ein Online-Algorithmus? Wann ist ein Algorithmus kompetitiv?
- Was hat das mit Approximationsalgorithmen zu tun?

Paging

- Was ist die Problemstellung? Welche Strategien gibt es?
- Warum vergleicht man sie mit offline Algorithmen?

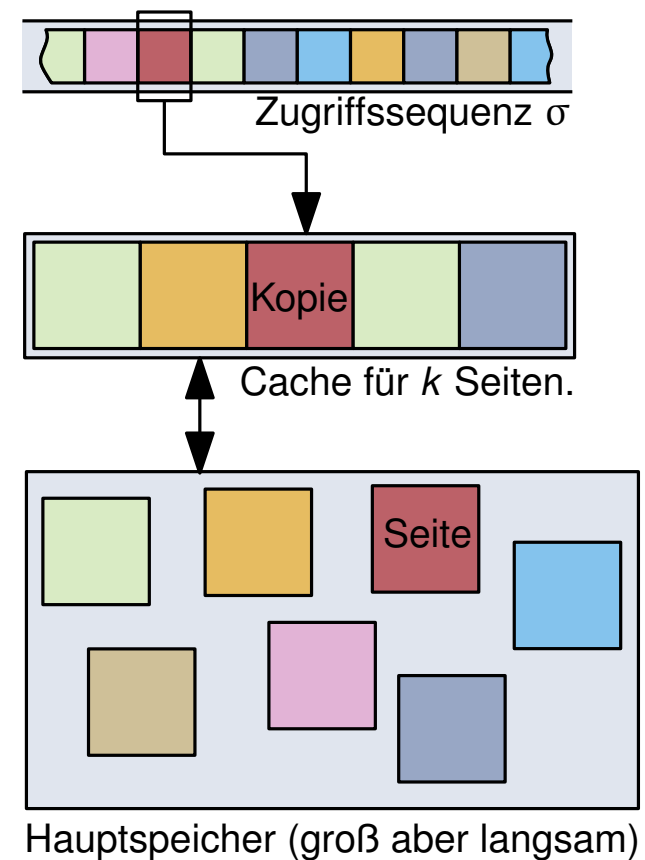


Grundlegendes

- Was ist ein Online-Algorithmus? Wann ist ein Algorithmus kompetitiv?
- Was hat das mit Approximationsalgorithmen zu tun?

Paging

- Was ist die Problemstellung? Welche Strategien gibt es?
- Warum vergleicht man sie mit offline Algorithmus?
- Warum kann man keine Gütegarantie besser als k erreichen?
- Was ist (h, k) -Paging? Wann ist ein Algorithmus konservativ?
- Inwiefern hilft (h, k) -Paging bei der Analyse?



Grundlegendes

- Wie misst man die Qualität eines parallelen Algorithmus?
- Was ist der speed-up und wann ist ein Algorithmus kostenoptimal?

Grundlegendes

- Wie misst man die Qualität eines parallelen Algorithmus?
- Was ist der speed-up und wann ist ein Algorithmus kostenoptimal?

Algorithmisches

- Wie kann man Summen (und andere binäre Operationen) parallel berechnen?

Grundlegendes

- Wie misst man die Qualität eines parallelen Algorithmus?
- Was ist der speed-up und wann ist ein Algorithmus kostenoptimal?

Algorithmisches

- Wie kann man Summen (und andere binäre Operationen) parallel berechnen?
- Was sind Präfixsummen und wie kann man sie parallel berechnen?

Grundlegendes

- Wie misst man die Qualität eines parallelen Algorithmus?
- Was ist der speed-up und wann ist ein Algorithmus kostenoptimal?

Algorithmisches

- Wie kann man Summen (und andere binäre Operationen) parallel berechnen?
- Was sind Präfixsummen und wie kann man sie parallel berechnen?
- Was ist List Ranking?

Grundlegendes

- Wie misst man die Qualität eines parallelen Algorithmus?
- Was ist der speed-up und wann ist ein Algorithmus kostenoptimal?

Algorithmisches

- Wie kann man Summen (und andere binäre Operationen) parallel berechnen?
- Was sind Präfixsummen und wie kann man sie parallel berechnen?
- Was ist List Ranking?
- Was ist die Idee zur parallelen Berechnung der Zusammenhangskomponenten eines Graphen? Wie kann das auf MST erweitert werden?

Die Vorlesung 21 zu *Algorithmen für externen Speicher* ist nicht klausurrelevant.

→ gehalten am 11.02.2014

Werbung

Lehrveranstaltungen nächstes Semester

- **Routenplanung (Master)**

Montag 14:00 – 15:30

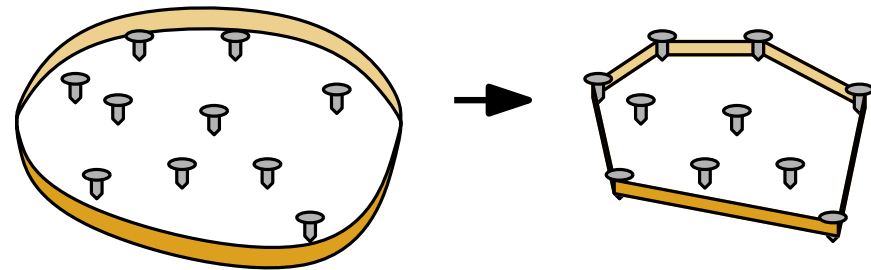
Mittwoch 11:30 – 13:00



- **Algorithmische Geometrie (Master)**

Vorlesung: Dienstag, 9:45 – 11:15

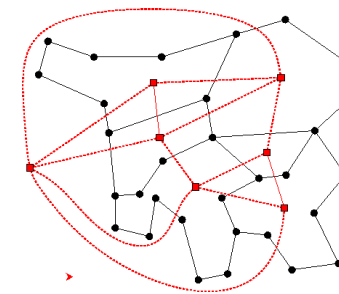
Übung: Mittwoch, 15:45 – 17:15



- **Algorithmen für planare Graphen (Bachelor)**

Dienstag 14:00 – 15:30

Mittwoch 14:00 – 15:30



i11www.iti.kit.edu/teaching