Fö. 10: Approximationsalgoritmer.

Fö. 11: Randomiserade algoritmer.

Fö. 12: Approximation + Randomisering.

Randomiserade algoritmer.

1. Grunder.
2. Min-Snitt.
3. Kontroll av matrismultiplikation.

Randomiserade algoritmer.

1. Grunder.
2. Min-Snitt — *betingade sannolikheter*.
3. Kontroll av matrismultiplikation — *överskott på vittnen*.

Grunder

Randomiserad algoritm: Använder slump och genererar en lösning som *förväntas* vara korrekt enligt något kriterium.

Grunder

Randomiserad algoritm: Använder slump och genererar en lösning som *förväntas* vara korrekt enligt något kriterium.

Vad menas med slump?

Grunder

Randomiserad algoritm: Använder slump och genererar en lösning som *förväntas* vara korrekt enligt något kriterium.

Vad menas med slump?

Vad menas med att en lösning förväntas vara korrekt?

Grunder

Randomiserade algoritmer är "nästan" lika bra som deterministiska.

Grunder

Randomiserade algoritmer är "nästan" lika bra som deterministiska.

Antag att en randomiserad algoritm svarar fel med sannolikhet 90%.

Kör en gång: $1 - 0.9 = 0.1$ chans att få korrekt svar.

Kör två gånger: $1 - 0.9^2 = 0.19$.

Kör tio gånger: $1 - 0.9^{10} \approx 0.65$.

Kör tjugo gånger: $1 - 0.9^{20} \approx 0.87$.

Kör hundra gånger: $1 - 0.9^{100} \approx 0.9999$.

Grunder

Randomiserade algoritmer kan (troligen) inte lösa **NP**-svåra problem väsentligen snabbare än deterministiska algoritmer.

Fördelar med randomiserade algoritmer.

1. Enkelhet.
2. Kan ge icke-trivial uppsnabbning.
3. Kraftfull metod i samband med approximation.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Tärning: $S = \{1, 2, 3, 4, 5, 6\}$.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Tärning: $S = \{1, 2, 3, 4, 5, 6\}$.

Väntevärde: $E(X) = \sum_{s \in S} p_s \cdot s$ där p_s är sannolikheten för utfall s.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Tärning: $S = \{1, 2, 3, 4, 5, 6\}$.

Väntevärde: $E(X) = \sum_{s \in S} p_s \cdot s$ där p_s är sannolikheten för utfall s.

Tärning: $E(X) = \sum_{i=1}^{6} \frac{1}{6} \cdot i = 3.5$.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Tärning: $S = \{1, 2, 3, 4, 5, 6\}$.

Väntevärde: $E(X) = \sum_{s \in S} p_s \cdot s$ där p_s är sannolikheten för utfall s.

Tärning: $E(X) = \sum_{i=1}^{6} \frac{1}{6} \cdot i = 3.5$.

Linearitet: $E(X_1 + \cdots + X_k) = E(X_1) + \cdots + E(X_k)$.

Grunder

Låt X vara en diskret stokastisk variabel med utfallsrum $S \subseteq \mathbb{N}$.

Tärning: $S = \{1, 2, 3, 4, 5, 6\}$.

Väntevärde: $E(X) = \sum_{s \in S} p_s \cdot s$ där p_s är sannolikheten för utfall s.

Tärning: $E(X) = \sum_{i=1}^{6} \frac{1}{6} \cdot i = 3.5$.

Linearitet: $E(X_1 + \cdots + X_k) = E(X_1) + \cdots + E(X_k)$.

Två tärningar: $E(X_1 + X_2) = E(X_1) + E(X_2) = 3.5 + 3.5 = 7$.

Exempel 1: Min-Snitt

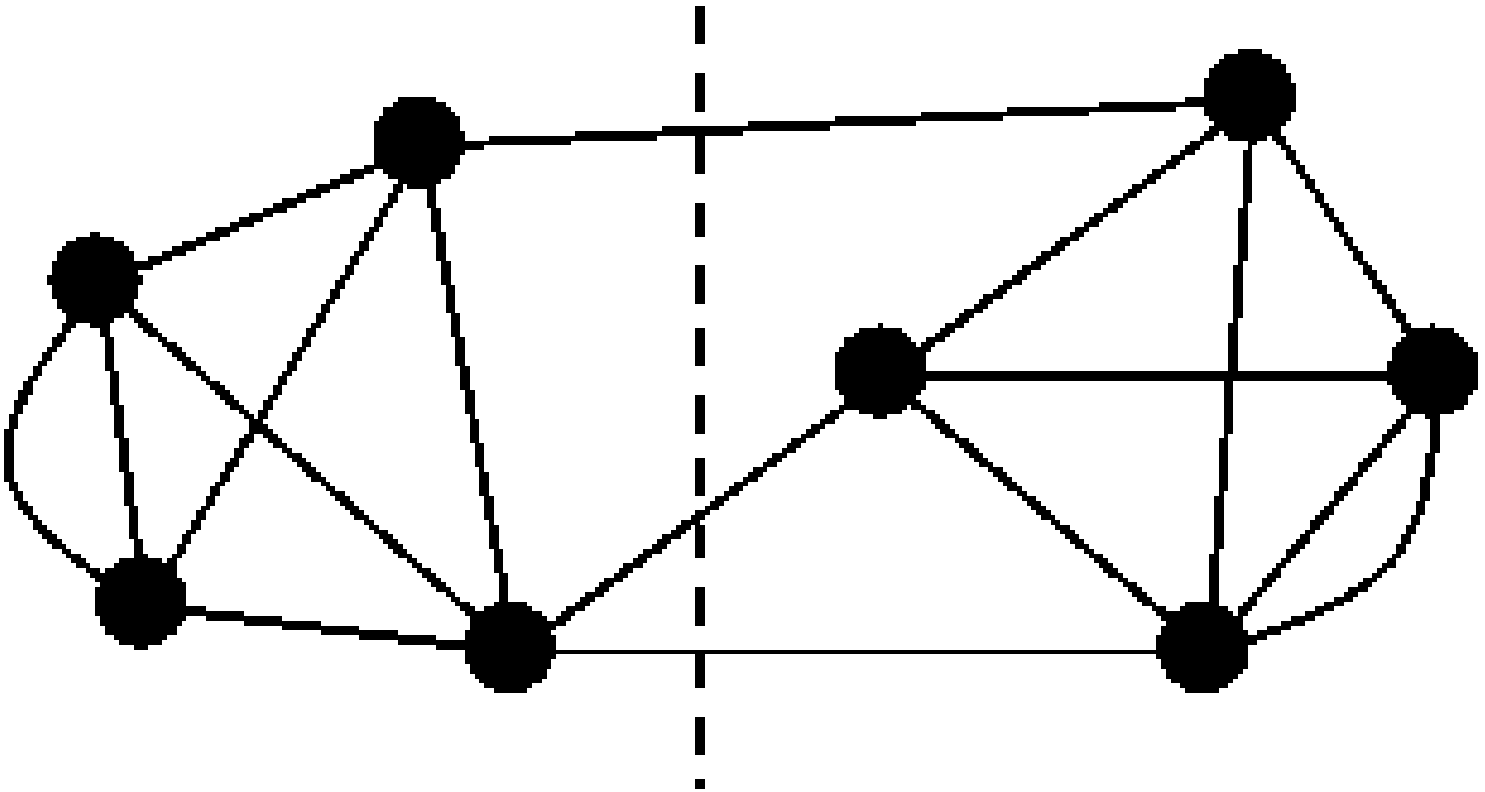
Min-Snitt

Problem. Min-Snitt.

Indata. Oriktad graf $G = (V, E)$.

Utdata. Delmängd $E' \subseteq E$ sådan att $G - E'$ inte är sammanhängande och E' har minimal storlek.

Min-Snitt



Min-Snitt

algoritm $K(G)$ där $G = (V, E)$ är en multigraf

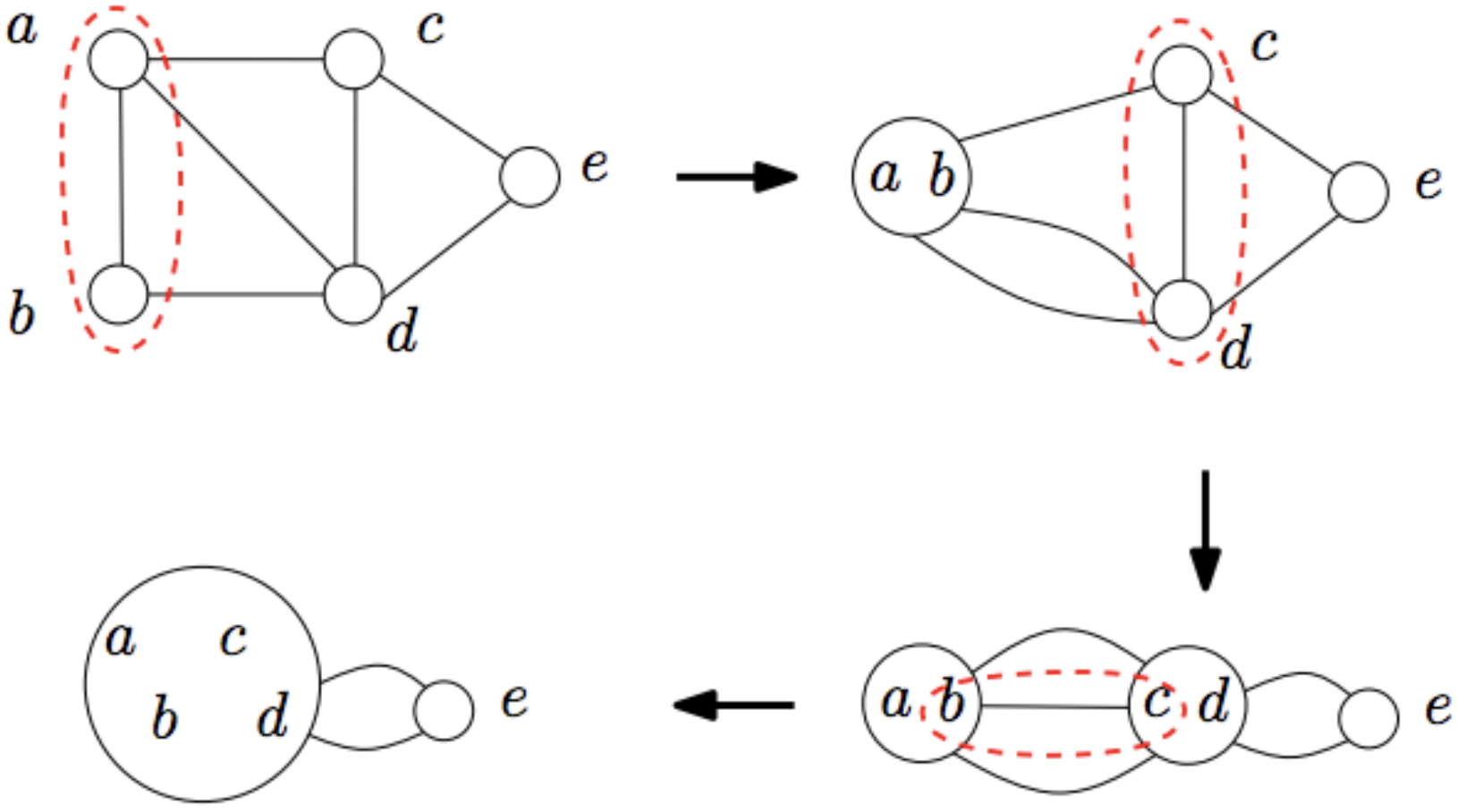
while $|V| > 2$

Välj slumpmässigt båge $(v_i, v_j) \in E$.

Uppdatera G genom att kontrahera v_i och v_j.

Bågarna mellan de två återstående noderna levereras som utdata.

Min-Snitt



Min-Snitt

Bevisidé.

Vad är sannolikheten att algoritmen "lyckas" i första varvet?

Min-Snitt

Bevisidé.

Vad är sannolikheten att algoritmen "lyckas" i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i andra varvet om den lyckas i första varvet?

Min-Snitt

Bevisidé.

Vad är sannolikheten att algoritmen "lyckas" i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i andra varvet om den lyckas i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i tredje varvet om den lyckas i tidigare varv?

Min-Snitt

Bevisidé.

Vad är sannolikheten att algoritmen "lyckas" i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i andra varvet om den lyckas i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i tredje varvet om den lyckas i tidigare varv?

⋮

Vad är sannolikheten att algoritmen "lyckas" i sista varvet om den lyckas i tidigare varv?

Min-Snitt

Bevisidé.

Vad är sannolikheten att algoritmen "lyckas" i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i andra varvet om den lyckas i första varvet?

Vad är sannolikheten att algoritmen "lyckas" i tredje varvet om den lyckas i tidigare varv?

⋮

Vad är sannolikheten att algoritmen "lyckas" i sista varvet om den lyckas i tidigare varv?

Utnyttja *betingade sannolikheter*.

Min-Snitt

Betingade sannolikheter.

$\Pr(A \mid B)$

"Sannolikhet för A om B har hänt".

Min-Snitt

Betingade sannolikheter.

$\Pr(A \mid B)$

"Sannolikhet för A om B har hänt".

Tärning: $\Pr_{x,y \in \{1,\ldots,6\}}(x + y \geq 11 \mid x = 6) = \frac{1}{3}$

Min-Snitt

Betingade sannolikheter.

$\Pr(A \mid B)$

"Sannolikhet för A om B har hänt".

Tärning: $\Pr_{x,y \in \{1,\ldots,6\}}(x + y \geq 11 \mid x = 6) = \frac{1}{3}$

Antingen gäller $y = 5$ eller $y = 6$.

Min-Snitt

Räkneregel.

$$
\begin{aligned}
&\Pr(H_1 \wedge \cdots \wedge H_k) = \\
&\Pr(H_1) \cdot \\
&\Pr(H_2 \mid H_1) \cdot \\
&\Pr(H_3 \mid H_1 \wedge H_2) \cdot \\
&\vdots \\
&\Pr(H_k \mid H_1 \wedge H_2 \wedge \cdots \wedge H_{k-1})
\end{aligned}
$$

Min-Snitt

Bevis.

Låt G vara en graf med n noder.

Min-Snitt

Bevis.

Låt G vara en graf med n noder.

Låt C vara ett min-snitt i G.

Min-Snitt

Bevis.

Låt G vara en graf med n noder.

Låt C vara ett min-snitt i G.

Händelsen H_i gäller om och endast om algoritmen *inte* väljer en båge i C under det i:te varvet.

Min-Snitt

Bevis.

Låt G vara en graf med n noder.

Låt C vara ett min-snitt i G.

Händelsen H_i gäller om och endast om algoritmen *inte* väljer en båge i C under det i:te varvet.

Om händelsen $H_1 \wedge \cdots \wedge H_{n-2}$ inträffar så returnerar algoritmen C vilket är ett korrekt svar.

Min-Snitt

Bevis.

Låt G vara en graf med n noder.

Låt C vara ett min-snitt i G.

Händelsen H_i gäller om och endast om algoritmen *inte* väljer en båge i C under det i:te varvet.

Om händelsen $H_1 \wedge \cdots \wedge H_{n-2}$ inträffar så returnerar algoritmen C vilket är ett korrekt svar.

Vi vill visa att $\Pr(H_1 \wedge \cdots \wedge H_{n-2})$ är "stor".

Min-Snitt

Låt $k = |C|$

Min-Snitt

Låt $k = |C|$

Det gäller att $|E| \geq \frac{kn}{2}$.

Min-Snitt

Låt $k = |C|$

Det gäller att $|E| \geq \frac{kn}{2}$.

Om detta inte gäller så finns en nod med grad $< k$ eftersom det inte finns tillräckligt med bågar för att ge varje nod grad $\geq k$.

Min-Snitt

Låt $k = |C|$

Det gäller att $|E| \geq \frac{kn}{2}$.

Om detta inte gäller så finns en nod med grad $< k$ eftersom det inte finns tillräckligt med bågar för att ge varje nod grad $\geq k$.

C är isåfall inte ett min-snitt.

Min-Snitt

$\frac{k}{kn/2}$ är högsta sannolikheten för att välja en båge i C: Det finns k bågar i C och minst $kn/2$ bågar i G.

$$\Pr(H_1) \geq 1 - \frac{k}{kn/2} = 1 - \frac{2}{n}.$$

Min-Snitt

$$\Pr(H_2 \mid H_1) \geq 1 - \frac{k}{k(n-1)/2} = 1 - \frac{2}{n-1}.$$

1. Hela C finns kvar i G pga. H_1.
2. En nod har försvunnit pga. kontraktionen.
3. Grafens minimala grad måste fortfarande vara minst k.

Min-Snitt

$$\Pr(H_i \mid H_1 \wedge \cdots \wedge H_{i-1}) \geq 1 - \frac{2}{n-i-1}.$$

Min-Snitt

$$\Pr(H_1 \wedge \cdots \wedge H_{n-2}) =$$

$$\Pr(H_1) \cdot$$
$$\Pr(H_2 \mid H_1) \cdot$$
$$\Pr(H_3 \mid H_1 \wedge H_2) \cdot$$
$$\vdots$$
$$\Pr(H_k \mid H_1 \wedge H_2 \wedge \cdots \wedge H_{n-2}) \geq$$

$$\prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \frac{2}{n(n-1)}$$

Min-Snitt

Sannolikheten är alltså $\approx 2/n^2$ att algoritmen hittar ett min-snitt.

Min-Snitt

Sannolikheten är alltså $\approx 2/n^2$ att algoritmen hittar ett min-snitt.

Ganska låg sannolikhet...

Min-Snitt

Sannolikheten är alltså $\approx 2/n^2$ att algoritmen hittar ett min-snitt.

Ganska låg sannolikhet...

Repetera $n^2/2$ gånger. Sannolikheten att inte hitta ett min-snitt blir då

$$\left(1 - \frac{2}{n^2}\right)^{n^2/2} < \frac{1}{e} \approx 0.36 \qquad \left[\lim_{x\to\infty}(1+1/x)^x = \mathrm{e}\right]$$

Med ytterligare repetitioner kommer man snabbt närmare sannolikheten 0.

Sammanfattning.

Algoritmen hittar ett min-snitt med hög sannolikhet.

Den går i polynomisk tid: en körning av algoritmen tar $O(n^2)$ tid och $O(n^2)$ upprepningar ger $O(n^4)$ total tid.

Exempel 2: Kontroll av matrismultiplikation.

Kontroll av matrismultiplikation

Problem. KMM.

Indata. Tre $n \times n$-matriser A, B och C.

Fråga. Är $C = A \cdot B$?

Kontroll av matrismultiplikation

Problem. KMM.

Indata. Tre $n \times n$-matriser A, B och C.

Fråga. Är $C = A \cdot B$?

Kan lösas i $O(n^3)$ tid med standardalgoritm för matrismultiplikation.

Kontroll av matrismultiplikation

Problem. KMM.

Indata. Tre $n \times n$-matriser A, B och C.

Fråga. Är $C = A \cdot B$?

Kan lösas i $O(n^3)$ tid med standardalgoritm för matrismultiplikation.

Kan lösas i $O(n^{2.3728596})$ tid med snabb matrismultiplikation.

Kontroll av matrismultiplikation

Problem. KMM.

Indata. Tre $n \times n$-matriser A, B och C.

Fråga. Är $C = A \cdot B$?

Kan lösas i $O(n^3)$ tid med standardalgoritm för matrismultiplikation.

Kan lösas i $O(n^{2.3728596})$ tid med snabb matrismultiplikation.

Kan lösas i $O(n^2)$ tid med randomiserad algoritm.

Kontroll av matrismultiplikation

Låt A, B, C vara givna $n \times n$-matriser.

Kontroll av matrismultiplikation

Låt A, B, C vara givna $n \times n$-matriser.

Definition. Låt $S \subseteq \{1, 2, \ldots, n\}$ och låt D vara en $n \times n$-matris. Definiera $\Sigma_S(D)$ som den n-vektor man får genom att addera de rader i D som indexeras av S.

Kontroll av matrismultiplikation

Låt A, B, C vara givna $n \times n$-matriser.

Definition. Låt $S \subseteq \{1, 2, \ldots, n\}$ och låt D vara en $n \times n$-matris. Definiera $\Sigma_S(D)$ som den n-vektor man får genom att addera de rader i D som indexeras av S.

Kan också ses som multiplikation med lämpligt vald 0/1-vektor v_S.

Kontroll av matrismultiplikation

$$M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 2 & 0 & 2 & 2 \\ 1 & 0 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

$$\Sigma_{\{2,4\}}(M) = [2,0,2,2] + [1,2,3,4] = [3,2,5,6].$$

Kontroll av matrismultiplikation

$$M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 2 & 0 & 2 & 2 \\ 1 & 0 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

$\Sigma_{\{2,4\}}(M) = [2,0,2,2] + [1,2,3,4] = [3,2,5,6].$

$v_{\{2,4\}} = [0,1,0,1].$

$[0,1,0,1] \cdot M = [2,0,2,2] + [1,2,3,4] = [3,2,5,6].$

Kontroll av matrismultiplikation

algoritm $F(A, B, C)$

välj $S \subseteq \{1, \ldots, n\}$ slumpmässigt (varje element 50% chans)

om $(\Sigma_S(A)) \cdot B = \Sigma_S(C)$ så svara "ja" och annars "nej".

Kontroll av matrismultiplikation

algoritm $F(A, B, C)$

välj $S \subseteq \{1, \ldots, n\}$ slumpmässigt (varje element 50% chans)

om $(\Sigma_S(A)) \cdot B = \Sigma_S(C)$ så svara "ja" och annars "nej".

Tidskomplexitet: $O(n^2)$.

Kontroll av matrismultiplikation

Påstående.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Kontroll av matrismultiplikation

Påstående.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Om $A \cdot B \neq C$ så svarar algoritmen "nej" med 50% chans.

Kontroll av matrismultiplikation

Påstående.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Om $A \cdot B \neq C$ så svarar algoritmen "nej" med 50% chans.

Om $A \cdot B \neq C$ så finns minst $2^n/2$ val av S som får algoritmen att svara "nej".

Kontroll av matrismultiplikation

Påstående.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Om $A \cdot B \neq C$ så svarar algoritmen "nej" med 50% chans.

Om $A \cdot B \neq C$ så finns minst $2^n/2$ val av S som får algoritmen att svara "nej".

Notera att man alltid kan lita på "nej"-svar. "Ja"-svar däremot är osäkra.

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

$(\Sigma_S(A)) \cdot B = \Sigma_S(C) \Leftrightarrow$

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

$(\Sigma_S(A)) \cdot B = \Sigma_S(C) \Leftrightarrow$

$(v_S \cdot A) \cdot B = v_S \cdot C \Leftrightarrow$

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

$$(\Sigma_S(A)) \cdot B = \Sigma_S(C) \Leftrightarrow$$
$$(v_S \cdot A) \cdot B = v_S \cdot C \Leftrightarrow$$
$$v_S \cdot (A \cdot B) = v_S \cdot C \Leftrightarrow$$

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

$$(\Sigma_S(A)) \cdot B = \Sigma_S(C) \Leftrightarrow$$
$$(v_S \cdot A) \cdot B = v_S \cdot C \Leftrightarrow$$
$$v_S \cdot (A \cdot B) = v_S \cdot C \Leftrightarrow$$
$$v_S \cdot (A \cdot B - C) = 0 \Leftrightarrow$$

Kontroll av matrismultiplikation

Observation.

Låt $D = A \cdot B - C$.

$$(\Sigma_S(A)) \cdot B = \Sigma_S(C) \Leftrightarrow$$
$$(v_S \cdot A) \cdot B = v_S \cdot C \Leftrightarrow$$
$$v_S \cdot (A \cdot B) = v_S \cdot C \Leftrightarrow$$
$$v_S \cdot (A \cdot B - C) = 0 \Leftrightarrow$$
$$v_S \cdot (D) = 0$$

Kontroll av matrismultiplikation

Fall 1. $A \cdot B = C$.

Kontroll av matrismultiplikation

Fall 1. $A \cdot B = C$.

$D = 0$.

Kontroll av matrismultiplikation

Fall 1. $A \cdot B = C$.

$D = 0$.

$v_S(D) = 0$ för alla val av S.

Kontroll av matrismultiplikation

Fall 1. $A \cdot B = C$.

$D = 0$.

$v_S(D) = 0$ för alla val av S.

$(\Sigma_S(A)) \cdot B = \Sigma_S(C)$ för alla S enl. tidigare observation.

Kontroll av matrismultiplikation

Fall 1. $A \cdot B = C$.

$D = 0$.

$v_S(D) = 0$ för alla val av S.

$(\Sigma_S(A)) \cdot B = \Sigma_S(C)$ för alla S enl. tidigare observation.

Algoritmen svarar "ja".

Kontroll av matrismultiplikation

Fall 2. $A \cdot B \neq C$.

Kontroll av matrismultiplikation

Fall 2. $A \cdot B \neq C$.

$D \neq 0$

Kontroll av matrismultiplikation

Fall 2. $A \cdot B \neq C$.

$D \neq 0$

Definition.
Antag att $S \subseteq \{1, \ldots, n\}$ och att den p:te raden i D innehåller minst ett element som inte är 0. Låt
$S' = S \cup \{p\}$ om $p \notin S$ och
$S' = S - \{p\}$ om $p \in S$.

Kontroll av matrismultiplikation

Fall 2. $A \cdot B \neq C$.

$D \neq 0$

Definition.
Antag att $S \subseteq \{1, \ldots, n\}$ och att den p:te raden i D innehåller minst ett element som inte är 0. Låt
$S' = S \cup \{p\}$ om $p \notin S$ och
$S' = S - \{p\}$ om $p \in S$.

Observation: $(S')' = S$.

Kontroll av matrismultiplikation

Viktig observation: $\Sigma_S(D)$ och $\Sigma_{S'}(D)$ kan inte båda vara 0.

Kontroll av matrismultiplikation

Viktig observation: $\Sigma_S(D)$ och $\Sigma_{S'}(D)$ kan inte båda vara 0.

Antag $\Sigma_S(D) = 0$ och $p \in S$. Då är $\Sigma_{S'}(D) = 0 - [\mathrm{rad}\ p] \neq 0$.

Kontroll av matrismultiplikation

Viktig observation: $\Sigma_S(D)$ och $\Sigma_{S'}(D)$ kan inte båda vara 0.

Antag $\Sigma_S(D) = 0$ och $p \in S$. Då är $\Sigma_{S'}(D) = 0 - [\mathrm{rad}\ p] \neq 0$.

Antag $\Sigma_S(D) = 0$ och $p \notin S$. Då är $\Sigma_{S'}(D) = 0 + [\mathrm{rad}\ p] \neq 0$.

Kontroll av matrismultiplikation

Viktig observation: $\Sigma_S(D)$ och $\Sigma_{S'}(D)$ kan inte båda vara 0.

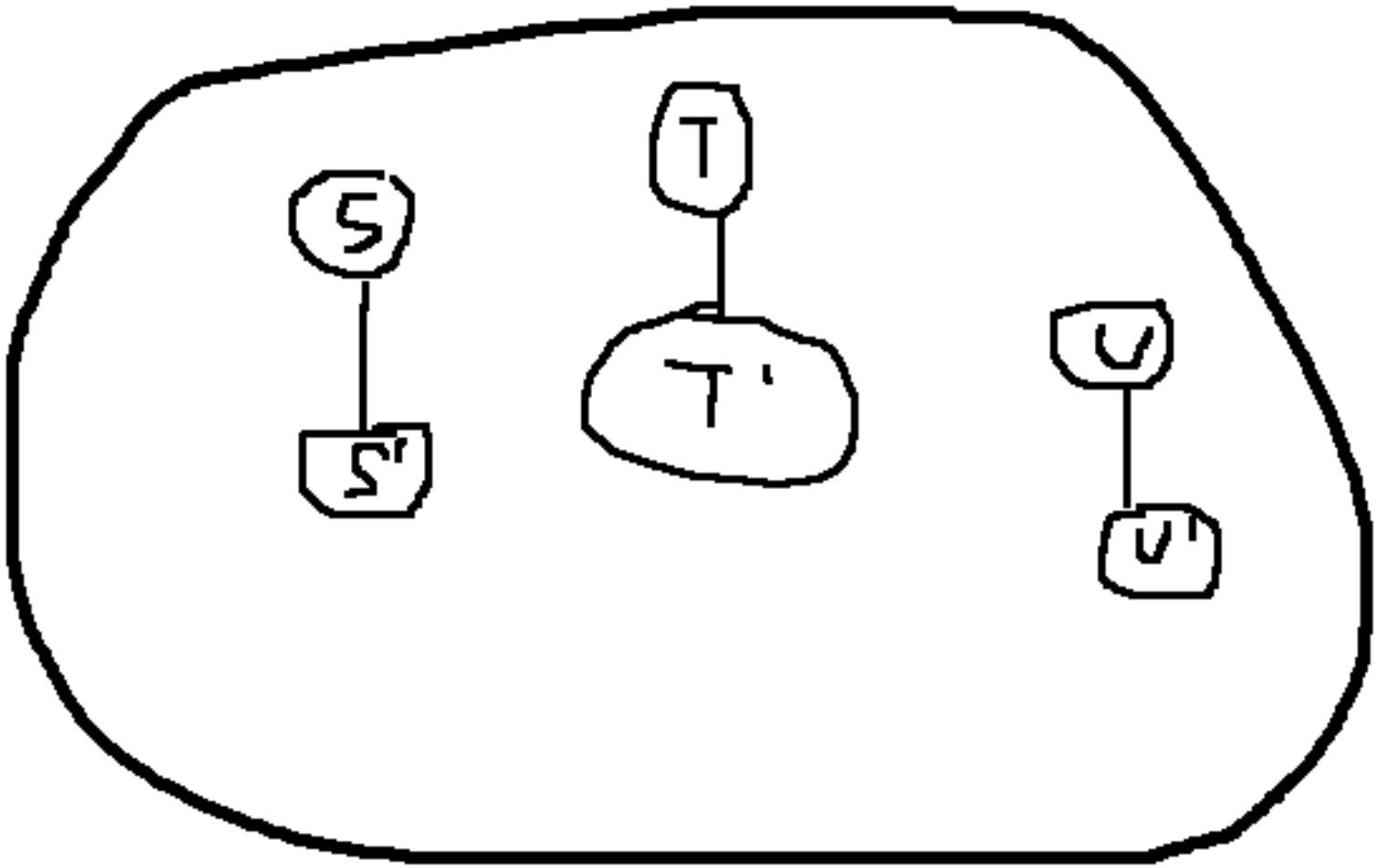
Antag $\Sigma_S(D) = 0$ och $p \in S$. Då är $\Sigma_{S'}(D) = 0 - [\mathrm{rad}\ p] \neq 0$.

Antag $\Sigma_S(D) = 0$ och $p \notin S$. Då är $\Sigma_{S'}(D) = 0 + [\mathrm{rad}\ p] \neq 0$.

Kontroll av matrismultiplikation

Slutsats: Varje S har en unik "kompis" S' och minst en av dessa ger ett resultat som är $\neq 0$.

Kontroll av matrismultiplikation



Kontroll av matrismultiplikation

Annorlunda uttryckt: Minst hälften av alla val av S gör att algoritmen svara "nej".

S är slumpmässigt vald så chansen att svara "nej" är minst 50%.

Kontroll av matrismultiplikation

Sammanfattning.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Kontroll av matrismultiplikation

Sammanfattning.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Om $A \cdot B \neq C$ så svarar algoritmen "nej" med 50% chans.

Kontroll av matrismultiplikation

Sammanfattning.

Om $A \cdot B = C$ så svarar algoritmen alltid "ja".

Om $A \cdot B \neq C$ så svarar algoritmen "nej" med 50% chans.

Algoritmen går i $O(n^2)$ tid.