به نام او

طراحی الگوریتم

۹۴/۱/۲۵

+ کوتاهترین مسیرها

کم وزن ترین مسیر

Dijkstra

- یال منفی

- دور منفی ← NP-کامل

- دور منفی ندارد ← Bellman-Ford

- کم وزن ترین مسیر بین همه راس ها : G, ω

$d[v,u]$

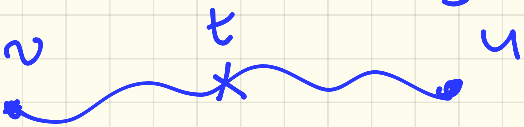
All pairs Shortest Paths

الگوریتم ۱:

اجرای dijkstra از هر راس

$O(V \times E \lg V)$

$\rightarrow O(n^3 \lg n)$



$d^i[v,u]$: کم‌وزن‌ترین مسیر v تا u ها از $\{1..i\}$
رد شود

A

i $i+1$ u

v

$$d^0 = A$$

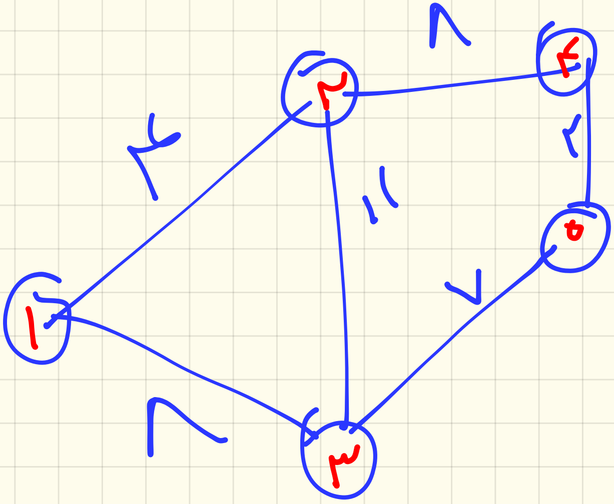
$$: d^{i+1} \leftarrow d^i$$

$$d^{i+1}[v,u] = \min\begin{cases} d^i[v,i+1] + d^i[i+1,u] \\ d^i[v,u] \end{cases}$$

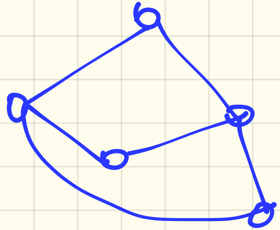
d^n : جواب

الگوریتم Floyd-Warshall:

```
FW(w)
  d = w
  for k : 0..n-1
      for i : 0..n-1
          for j : 0..n-1
              d[i,j] = min(d[i,j], d[i,k] + d[k,j])
```


۱
۲
۳
۴
۵
۴
۸
۱۱
۸
۷
۶

مسئله :



$\sum_{v,u} d_{vu}$ = نارضایتی

C_i : هزینه نابود کردن i

تصمیم : k : شهر را k تا نابود

$\max$ نارضایتی $- \sum_{i=1}^{k} c_i$

$x^* = 0$

for $t : 0 \dots n-1$

$\quad d \leftarrow FW(G_{t \dots n-1})$

$\quad x = \sum d - c_0 \dots c_{t-1}$

$\quad x^* = \max(x, x^*)$

$$\left.\right\} O(n^4)$$

۵۸٪ :

$A[v, u]$:

کمترین فاصله از v به u چیست؟

: $\underline{\underline{G}}$ —

به نام او

طراحی الگوریتم

۹۴/۱/۳۰

+ درخت فراگیر کمینه

درخت فراگیر کمینه :

۱- همه به هم متصل ← درخت فراگیر

۲- هزینه کمینه

الگوریتم پوستی ،

هر دفعه یک دور انتخاب می‌کنیم

حذف سنگین‌ترین یال   ۹   ۵

الگوریتم دهقانی

یک رأس انتخاب می‌کنیم (v)   ۲   ۸

درخت فراگیر کمینه در G-v ← T

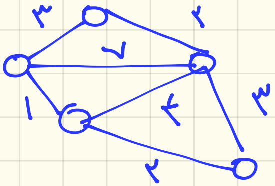
جواب = (T به v یال کمینه) + T

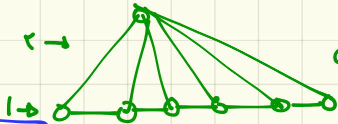
الگوریتم ریاضی X

هر رأس v :

T_v = درخت کوتاه‌ترین مسیر از v

جواب = کمینهٔ T_v ها





مثال نقض ⇐

ح ۲: حذف سنگین‌ترین یال که گراف را ناهمبند نمی‌کند

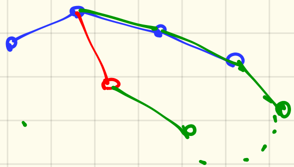
همه   ۵

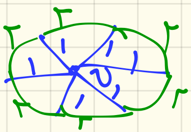
Kruskal

ح ۳: اضافه کردن سبک‌ترین یالی که دور ایجاد نکند   ۱۱   ۲

ح ۴: اضافه کردن سبک‌ترین یال با حفظ همبندی

Prim   ۷   ۵





به نام او

طراحی الگوریتم

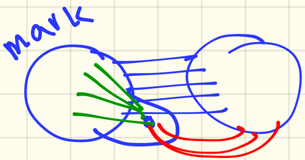
۹٤/۲/۱

+ پیاده سازي درخت کمینه

Prim(G, v)

mark ← 0
mark[v] ← true

for u in مج v
    d.insert(w(v, u), u, v)



for i ← 0 to n-۲ → $O(n^3)$
    x ← ∞
    for u in V
        if mark[u]
            for s in V
                if !mark[s]
                    if w[u][s] < x
                        x ← w[u][s]
                        v′ ← s
                        u′ ← u
    mark[v′] ← true
    print ("u′ → v′")

while !d.empty() {   d : heap
    (w′, v′, u′) ← d.pop() → $O(E \lg V)$
    if !mark[v′]
        mark[v′] ← true
        for u مج v′:
            d.insert((w(u, v′), u, v′))

الگوریتم $O(V^2)$ [illegible]
ارائه دهید

به نام او

طراحی الگوریتم

۹٤/۲/٦

MST - درخت فراگیر کمینه

Kruskal : افزودن یال کمینه

$O(E \lg E) = O(E \lg V)$

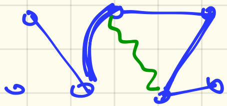
$O(E \lg E + E \lg^* V)$

Prim : گسترش

$O(E \lg V)$

$O(V^2)$

$O(E + V \lg V)$

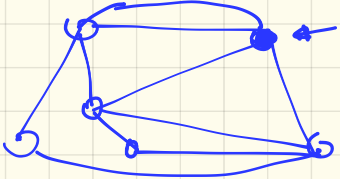


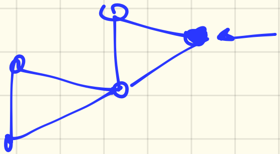
q: دو رأس در یک مجموعه‌اند ?

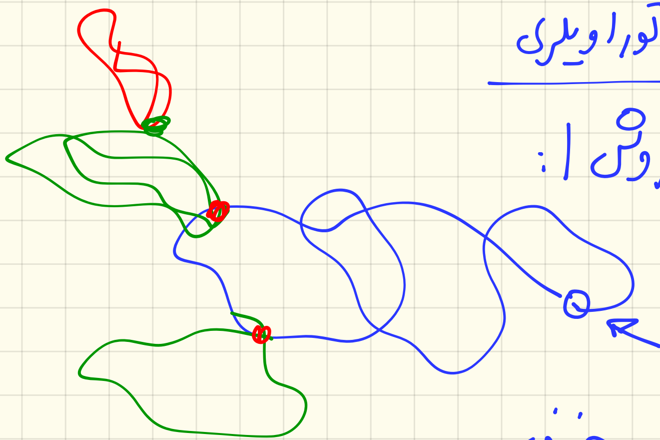
q: ادغام ۲ مجموعه

تور اویلری

روش ۱:

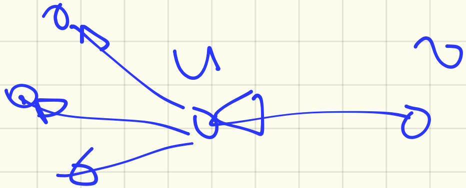


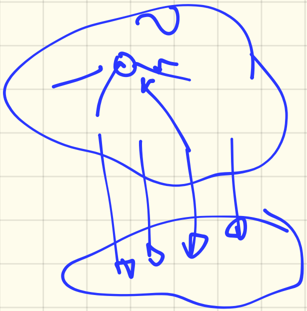


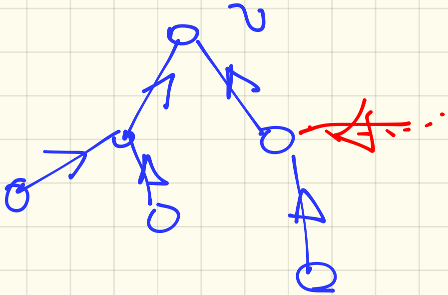


حذف

الگوریتم ۲: dfs روی یال‌ها







الگوریتم غیر شهودی

درخت فراگیر ذخیره

حرکت دلبخواهی

اگر یه از درخت بالا آمدیم

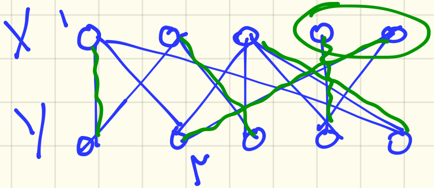
به نام او

طراحی الگوریتم

۹۴/۲/۸

+

تطابق بیشینه: بیشترین تعداد آدم را به کار نسبت بدهیم

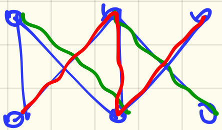


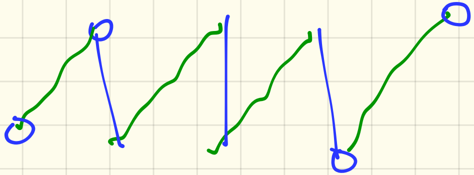
M: - همسایه نباشند
- بیشینه

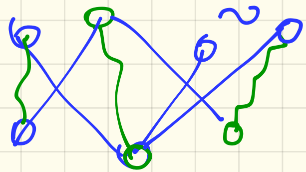
شرط لازم و کافی برای وجود تطابق کامل: (قضیه هال)

$\forall S \subseteq X : |S| \le |N(S)| \iff$ بتوان X را منطبق کرد

الگوریتم برای: ... 2^n x



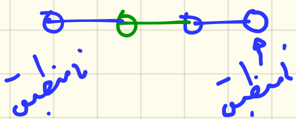




v ∈ X که منطبق شده

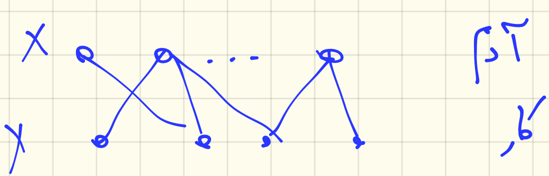
آبی ↓ سبز ↑

- مسیر افزایشی:



مسیر افزایشی نداشتن $\iff$ تطابق بیشینه

بهترین + آقازاده!



به نام او

طراحی الگوریتم

۹۴/۲/۱۳

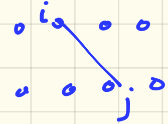
+

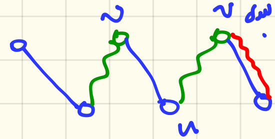
G [i][j]
mark
umatch[ ] ← -1

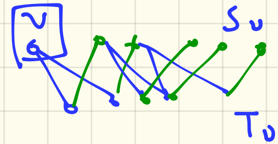
```
mdfs (v)
  if v == -1  return true
  if mark[v]  return false
  mark[v] ← true
  for u : همسایه‌ی v
     if (mdfs (umatch[u]))
        umatch[u] = v
        return true
  return false

for v ∈ رئوس
   mark[ ] = false
   mdfs(v)
```

تطابق





① 

$S \subset \{v\} \cup V_i$

② مسیرهای افزایشی دیگر

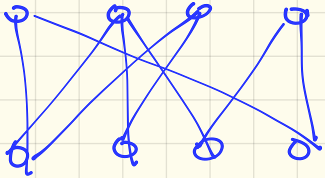
$|N(S)| < |S|$

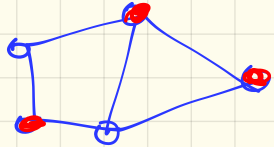
③ اتفاق بد 

$O(EV)$

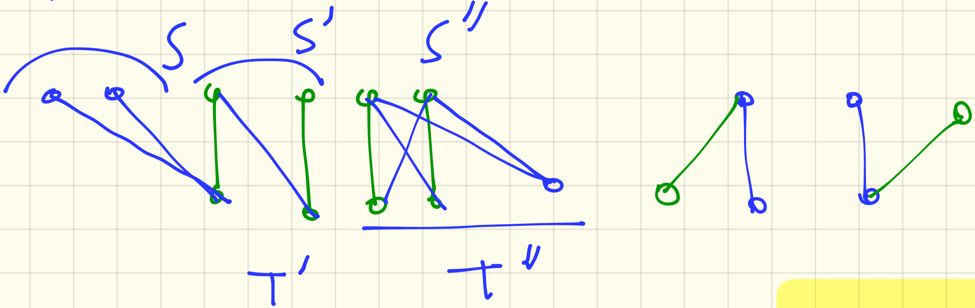
پوشش رأسی

گراف ۲ بخشی





|پوشش رأسی کمینه| = |تطابق بیشینه|



به نام او

طراحی الگوریتم

۹۴/۲/۲۰

+تبدیل فوریه سریع

ذخیره سازی چندجمله‌ای‌ها

چندجمله‌ای درجه n

۱- نگهدار ضرایب

~~۲- نگهداری ریشه‌ها و ضریب x^n~~

۳- نگهدار مقدار چندجمله در $n+1$ نقطه متمایز

$$x_0, \dots, x_n \to y_0, \dots, y_n \to P \qquad z_0, \dots, z_n \to Q$$

$$\prod_{i \neq j} \frac{x - x_i}{x_j - x_i} = P_j \qquad \sum_{j=0}^{n} y_j P_j$$

$$P(x) = a_n x^n + \sim + a_0$$

$$Q(x) = b_n x^n + \cdots + b_0$$

$$P.Q(x) = c_{2n} x^{2n} + \cdots + c_0$$

$$c_i = \sum_{j=0}^{i} a_j b_{i-j} \qquad O(n^2)$$

$$x_{n+1}, \dots, x_{2n}$$

$$(x_0, \dots, x_n), \ (y_0 z_0, \dots, y_{2n} z_{2n})$$

مقدار یک چندجمله‌ای در نقاط x_{2n} و ... و x_1 (که می‌توانند به انتخاب ما باشند) داده شده است

محاسبه سریع چندجمله‌ای در نقاط دیگر؟

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{2n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{2n-1} \\ \vdots & & & & \\ 1 & x_{2n} & \cdots & & x_{2n}^{2n-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{2n-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_{2n} \end{bmatrix}$$

$x_2 = x_1^2 \qquad 1 \quad x_1^2 \quad x_1^4 \cdots x_1^{2n} \quad x_1^{2n+2} \cdots x_1^{4n-2}$

$x_2 = -x_1$

$P(x) = a_0 + a_1 x + \cdots + a_{2n-1} x^{2n-1} \qquad P(-x) = P_e(x) - xP_o(x)$

$= P_e(x^2) + xP_o(x^2)$

$a_1 + a_3 x^2 + a_5 x^4 + \cdots + a_{2n-1} x^{2n-2}$

$a_0 + a_2 x^2 + a_4 x^4 + \cdots + a_{2n-2} x^{2n-2}$

محاسبه ۲ چندجمله‌ای درجه n ← محاسبه ۲ چندجمله‌ای درجه n + n ...

$$\begin{bmatrix} 1 & x_1 & \cdots & x_1^{2n-1} \\ \vdots & & & \\ 1 & x_n & \cdots & x_n^{2n-1} \\ 1 & (-x_1) & \cdots & (-x_1)^{2n-1} \\ \vdots & & & \\ 1 & (-x_n) & \cdots & (-x_n)^{2n-1} \end{bmatrix}$$

$$P(x) = \underbrace{P_e(x^2) + x P_o(x^2)}_{\text{درجه } n-1}$$

$x_1^2, \ldots, x_n^2$ $\qquad$ $x_{n/2+1} = i x_1$

$x_i = \omega^i$ $\qquad$ ω، ریشه ۲n ام واحد:

اولیه

n توانی از ۲

ω: ریشه اولیه nام واحد

$$\omega = e^{\frac{2\pi i}{n}}$$

$$1 \notin \{\omega^1, \dots, \omega^{n-1}\} \qquad \omega^n = \omega^0 = 1$$

$$x_i = \omega^i$$

$$(x_i)^{2^k} = \left(x_{i+\frac{n}{2^k}}\right)^{2^k}$$

$$x_i^2 = \left(x_{i+\frac{n}{2}}\right)^2$$

مثال: $0 + x + 2x^2 + 3x^3 + 4x^4 + 5x^5 + 6x^6 + 7x^7$

ω ریشه ۸ام واحد

$P_{a_0,\dots,a_n}(x_0,\dots,x_n)$ $\qquad$ $P_{0,1,\dots,7}(1,\omega,\dots,\omega^7)$

$$= P_{0,2,4,6}(1,\omega^2,\omega^4,\omega^6) + \omega P_{1,3,5,7}(1,\omega^2,\omega^4,\omega^6)$$

$$\underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega & \cdots & \omega^{n-1} \\ \vdots & & & \\ 1 & \omega^{n-1} & \omega^{2(n-1)} \cdots & \omega^{(n-1)^2} \end{bmatrix}}_{V(\omega)} \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_{n-1} \end{bmatrix}$$

تبدیل فوریه (FFT)

$$V(\omega)^{-1} = V\left(\frac{1}{\omega}\right) \times \frac{1}{n}$$

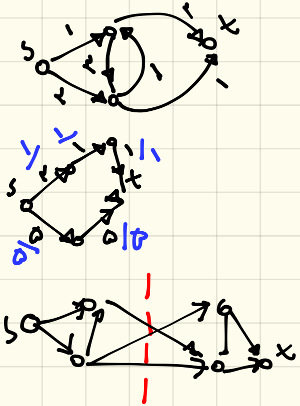
ω^{-1} : ریشه nام واحد

به نام او

طراحی الگوریتم

۹۴/۲/۲۷

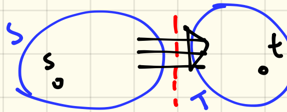
+شار



شار بیشینه (max flow)

اندازه: لیتر بر ثانیه‌ای که توانستیم رد کنیم

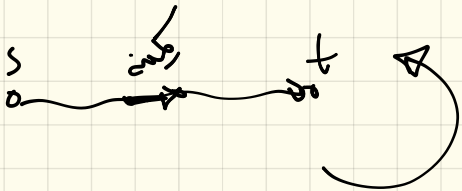
1. $f(e) \le c(e)$
2. $\forall v \neq s,t \quad \sum_{e:\ \text{ورودی } v} f(e) = \sum_{e:\ \text{خروجی } v} f(e)$
3. $f \geq 0$

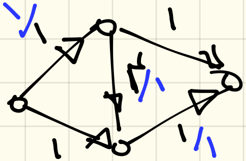


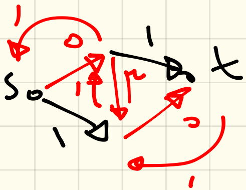
برش کمینه (min cut)

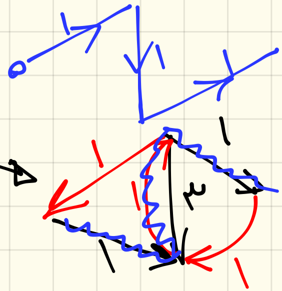
اندازه برش: جمع وزن یال‌های از S به T

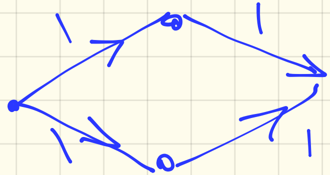
شار بیشینه $\geq$ برش کمینه

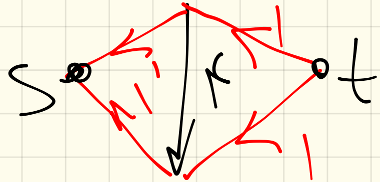



1
1
μ
1
1


s
t
1
0
1
μ
1
0
1


1
1
1
1
μ
1
1


1
1
1
1

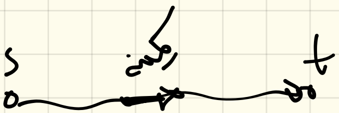

s
t
μ
1
1

به نام او

طراحی الگوریتم

۹۴/۲/۲۹

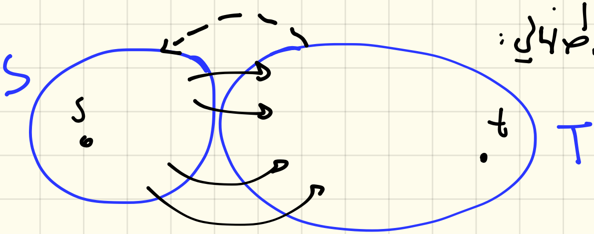
+شار

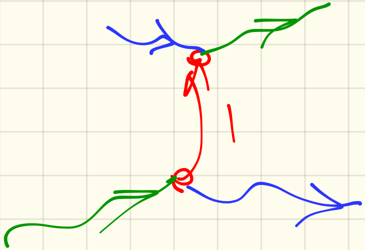


کم کردن ظرفیت ها

+ یال های قرمز

شرط پایان:





$f = c$

اندازه این برش = اندازه شار از S به T

شار بیشینه = برش کمینه

مثال

دانشجوی دکتری ۵ ظرفیت $\xleftrightarrow{\text{علاقه به همسانی}}$ استاد ظرفیت خالی

به نام او

طراحی الگوریتم

۹۴/۳/۳

+ پیچیدگی محاسبه

P

کارا ← زمان اجرای الگوریتم برحسب اندازه ورودی چندجمله‌ای باشد

$L \subset U$

مطلوب (جواب مثبت)

مسئله تصمیم‌گیری $\Leftrightarrow$ زبان L

$L_1 \subseteq U_1$ $\quad$ $L_2 \subseteq U_2$

$U_2 = \{$بله و خیر$\}$

$L_2 = \{$بله$\}$

$f: U_1 \to U_2$

$x \in L_1 \Leftrightarrow f(x) \in L_2$ $\quad$ قابل محاسبه در زمان چندجمله‌ای

L_1 سخت‌تر از L_2

$$\begin{array}{c} L_2 \\ \uparrow \\ L_1 \end{array}$$

L_1 سخت‌تر از L_2

$L_2 = \underbrace{f(L_1)} \cup (L_2 \setminus f(L_1))$

مسئله P: الگوریتم چندجمله‌ای برای تشخیص L

گزاره: سخت‌تر بودن دارای خاصیت تعدی است.

$$\begin{tikzcd} L_3 \\ L_2 \arrow[u] \\ L_1 \arrow[u] \arrow[uu, bend left] \end{tikzcd}$$

$$L_1 \Longleftrightarrow L_2 \qquad \begin{tikzcd} L_1 \\ L_2 \arrow[u] \end{tikzcd} \qquad \begin{tikzcd} L_2 \\ L_1 \arrow[u] \end{tikzcd}$$

ما الگوریتم غیرقطعی زبان L را می‌پذیرد اگر برای هر x مقداردهی‌ای از nd_انتخاب‌ها وجود داشته باشد که الگوریتم جواب مثبت دهد اگر و فقط اگر $x \in L$.

$$(x \wedge y) \vee z$$

به نام او

طراحی الگوریتم

۹۴/۳/۵

+ پیچیدگی محاسبه

مسئله SAT :

- الگوریتم قطعی ← ردهٔ پیچیدگی P
- الگوریتم غیر قطعی : $M(x,u)=1 \iff \exists u :$ $=1$ جواب x
- ردهٔ پیچیدگی : NP

الگوریتم غیر قطعی چندجمله ای.

- $P \stackrel{?}{=} NP$

- $SAT \in NP\text{-}C$:

به ازای هر مسئله L در NP ← f_L $x \in L \iff f_L(x) \in SAT$

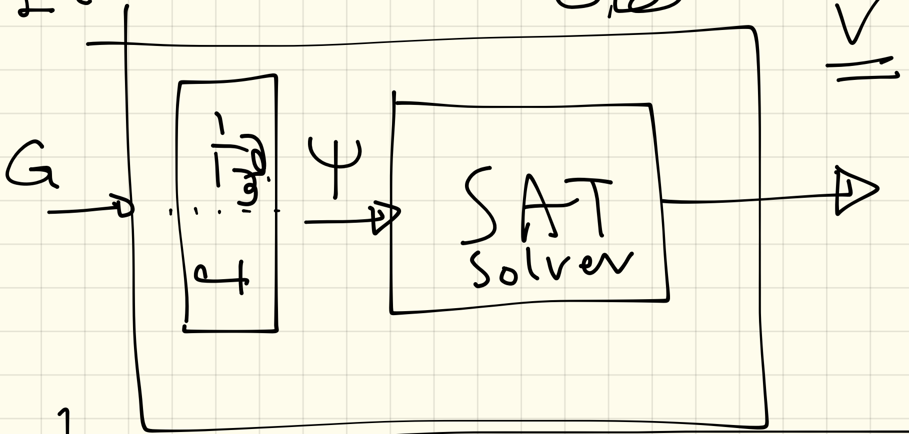
- مسئله هایی که از NP-C ها سخت ترند :

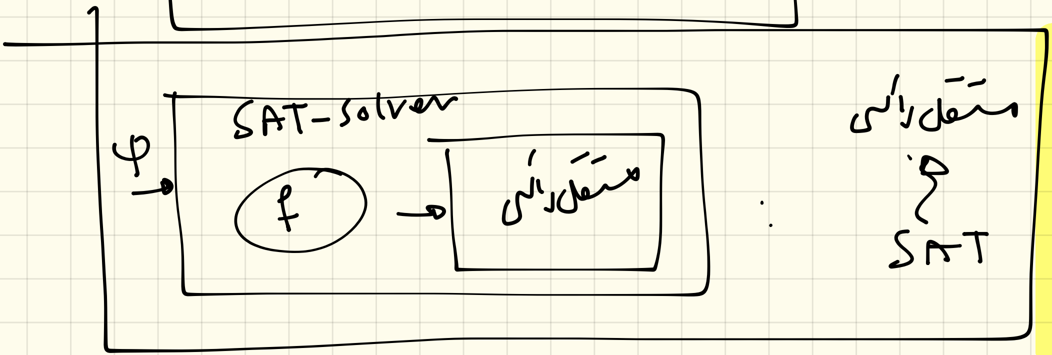
$SAT \Leftrightarrow L \in NP$

$x_1, \dots, x_n$

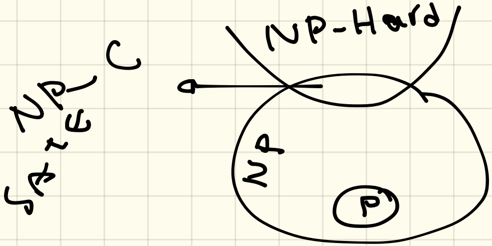
$(\bar{x}_1 \vee x_2) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (\quad)$

تطابق





مستقل راسی $\Leftrightarrow$ SAT



- NP-Hard: سخت‌تر از همه NP

SAT ∈ NP-Hard

- قضیه:

۳-SAT
↑
SAT

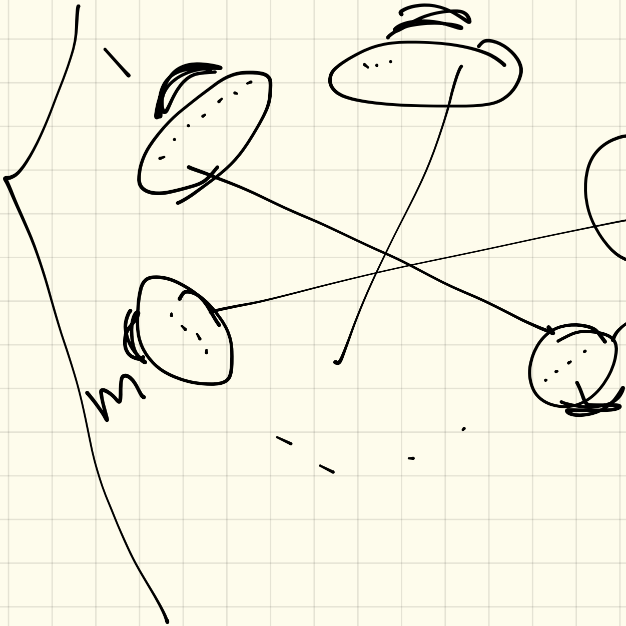
- تعریف: مسئله مجموعه مستقل راس: ⟨G,k⟩

↑

- قضیه: ۳-SAT: تعریف ۷۵

f(   )
↑
۳-SAT

$(\bar{x}_1 \vee x_2 \vee \bar{x}_3) \wedge (\quad) \wedge (\quad) \wedge (\quad)$ $\qquad x_1, \dots, x_n$



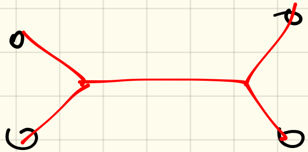
$(\bar{x}_1 \vee x_3 \vee \bar{x}_9)$

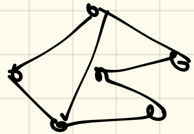
x_1	x_3	x_9
T	T	T
T	T	F
~~T~~	~~F~~	~~T~~
T	F	F
F	T	T
F	T	F
F	F	T
F	F	F

m

$\varphi: SAT \rightarrow G, k$: متناظر با متغیرهای گزاره‌ها ← راس‌ها ← مجموعه مستقل m-تایی

$\langle G, k \rangle \rightarrow \varphi: sat \rightarrow \checkmark$





TSP