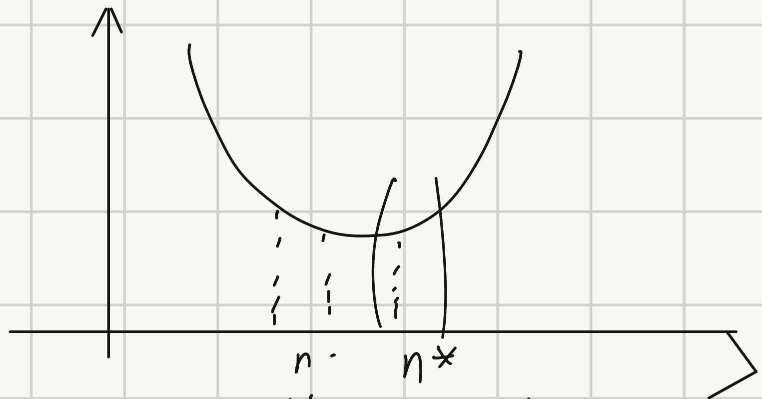
Local search

$f(n) = (an-b)^2$



this part is visible, near n.

find $\underset{n \in \mathbb{Z}}{\operatorname{argmin}} f(n)$

Optimization problem. (minimization)

$\mathcal{E} = \{ s \mid s \text{ is a feasible set} \}$

cost : $c = \mathcal{E} \to \mathbb{R}$

find $\underset{s \in \mathcal{E}}{\operatorname{argmin}} \ c(s)$

Local search :

1. pick a solution s from $\mathcal{E}$.
2. while s has a better neighbor s' $(c(s') < c(s))$
3. $s := s'$

Vertex Cover Problem

Given a $G = (V, E)$

find a minimum $\underbrace{\text{vertex cover } S}$.

$S \subseteq V$, s.t. every $e \in E$ has at least one endpoint in S.

$\mathcal{C} = \{S \mid S \text{ is a vertex cover}\}$

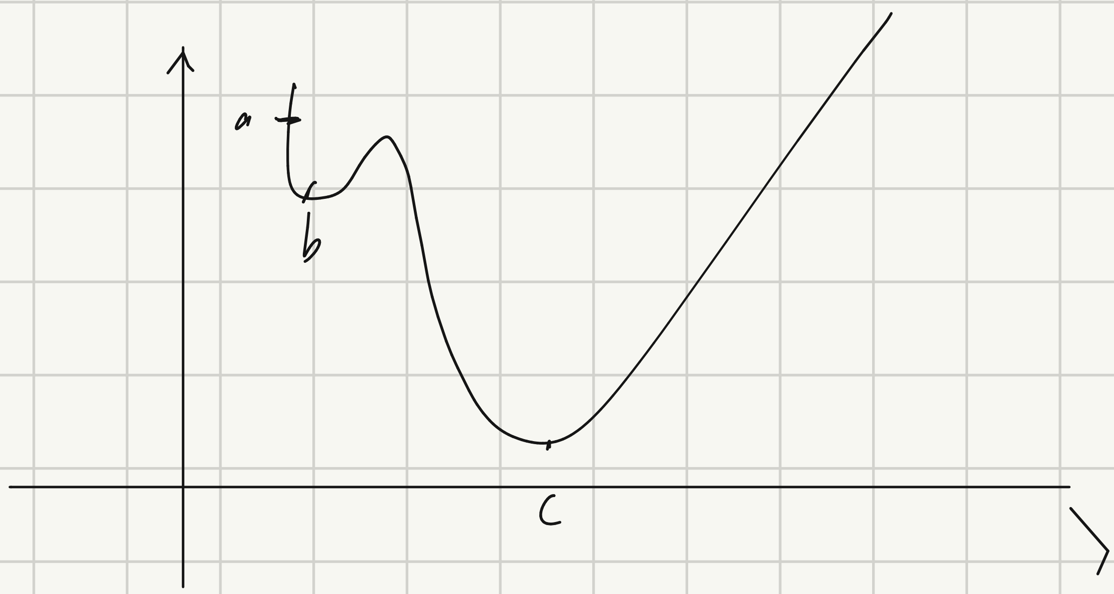
$C(S) = |S|$

neighborhood of S.

$N(S) = \{S' \mid S'$ is a vertex cover and S' can be obtained from S by adding or deleting a single node $\}$

LSVC (V, E)

1. $S = V$.
2. if $S - \{u\}$ is a vertex cover for some $u \in S$
3. $\quad S = S - \{u\}$



a: start of local search

b: end at this

but b is local optimum

c is global optimum

Metropolis algorithm

1. let K, T be two constant.
2. pick a solution S from $\mathcal{C}$
3. while true:
4. $\quad$ randomly pick a solution S' from $N(S)$

5. if $c(s') < c(s)$

6. $s := s'$

7. else // $c(s') \geq c(s)$

8. set $s := s'$ with probability $e^{-\frac{\Delta c}{KT}}$

($\Delta C = c(s') - c(s)$)

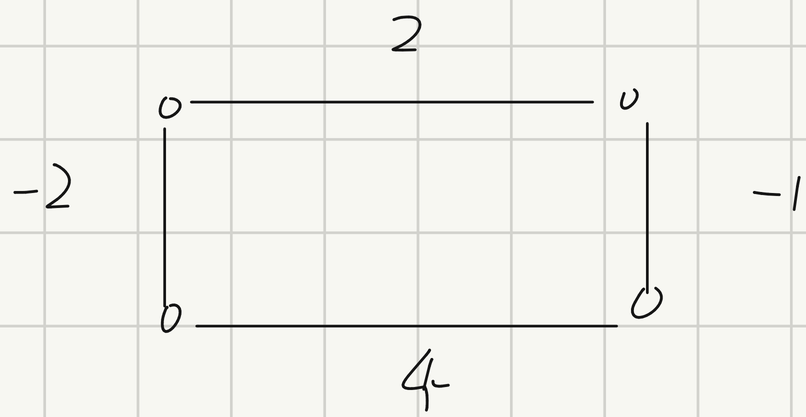
9. break when certain condition holds.

Simulated Annealing

gradually decreasing T

Hopfield Network Problem

Input. $G = (V, E)$ with edge weight $w: E \to \mathbb{Z}$



$s: V \xrightarrow[\text{value}]{\text{set}} \{-1, +1\}$

$\searrow$ configuration

Given a configurations, $e = (u, v)$ is good edge if

target: <1> $W_e > 0$, $s(u) \neq s(v)$ $\quad W_e \cdot S_v \cdot S_u < 0$

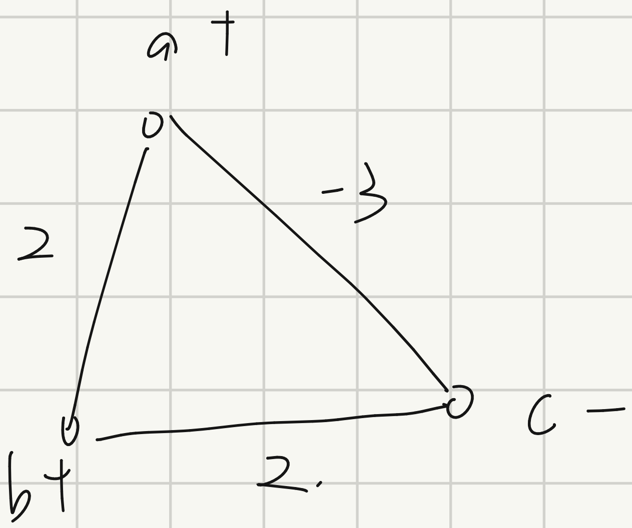
<2> $W_e < 0$, $s(u) = s(v)$ (same signal)

bad otherwise.

objective. 1. max $\sum_{e \text{ is good edge}} |W_e|$

objective 2. u maximize $\sum_{\substack{\text{good edge} \\ e \text{ incident} \\ \text{to } u}} |W_e| = \text{maximise } \mho(u)$

eg:



good: $\{bc\}$
bad $\{ab, ac\}$

$\mho(c) = 2$

$\to$ $c \to +$ good $\{ac\}$
bad $\{ab, bc\}$

$\mho(c) = 3.$

Given a configuration s, a node u is satisfied if

$$\sum_{\substack{\text{good edge} \\ e \text{ incident} \\ \text{to } u}} |W_e| \geq \sum_{\substack{\text{bad edge} \\ e \text{ incident} \\ \text{to } u}} |W_e|$$

A configuration s is stable if every node u is satisfied

State-flipping:

1. pick an arbitrary configuration s
2. while some node u is not satisfied
3. flip the state of u.
4. return S

local search to maximize objective 1

i.e. $\Phi(s)$

To prove the algorithm above works, that is to say the iterations isn't infinite.

We define $\Phi(s) = \sum_{e \text{ is good}} |W_e| \rightarrow$ objective 1

$$S \xrightarrow{\text{flip } u} s'$$

$\Phi(s)$ $\qquad$ $\Phi(s')$

$$\Phi(s') = \Phi(s) - \sum_{\substack{\text{good edge} \\ \text{incident to } u}} |W_e| + \sum_{\substack{\text{bad edge} \\ \text{incident to } u}} |W_e|$$

$\Rightarrow \Phi(s') > \Phi(s)$ (it's the goal why we flip)

$\Rightarrow \Phi(s') \geqslant \Phi(s) + 1$

$$\Phi(s) \leqslant \sum_e |W_e| = W$$

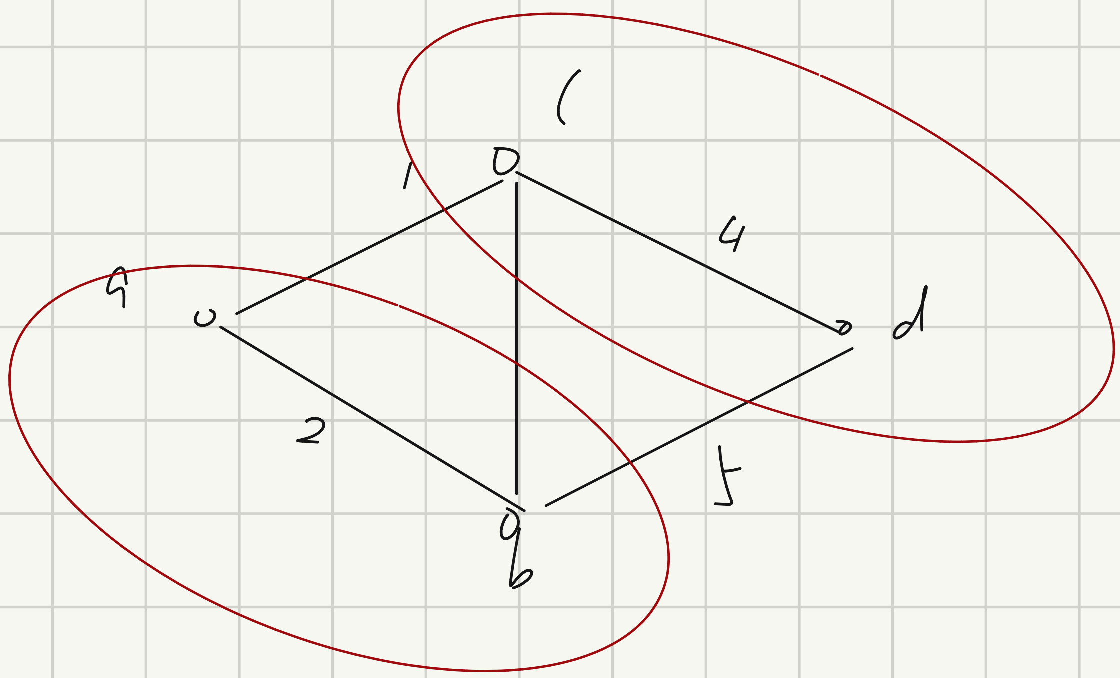
Each flip add one to $\Phi(s)$, and $\Phi(s)$ has an upper bound, so its a finite iterations. so it works

Maximum Cut (NP - hard)

Given a undirected graph $G = (V, E)$ with edge weight $w : E \to \mathbb{Z}^+$. A cut (A, B) is a partition of V into two none-empty subsets

$$\delta(A,B) = \{ (u,v) \in E \mid u \in A,\ v \in B \}$$

$$W(A,B) = \sum_{e \in \delta(A,B)} W_e$$



Input : an edge weighted graph $G = (V, E)$

Output : a maximum cut

$\Rightarrow$ A special case of Hopfield with $W_e > 0$ for all e
(e is a cut edge $\Leftrightarrow$ e is a good edge)

State - flip - Max - Cut :

1. pick an arbitrary cut (A, B)
2. while some node u is not satisfied.

$$\left(\sum_{\text{cut edge}} W_e < \sum_{\text{non-cut edge}} W_e \right)$$

incident to u incident to u

3 flip the memship of u

$\underbrace{\text{local optimum}}_{\Rightarrow \text{2-approximation.}}$ in $O(W)$ iterations. $\nearrow$ input $\log W$ $\searrow$ pseudo-poly

(A,B) is stable

for $u \in A$,

$$\sum_{\substack{e=(u,v)\in E \\ v\in A}} \le \sum_{\substack{e=(u,v)\in E \\ v\in B}} W_e$$

$$2\sum_{\substack{(u,v)\in E \\ u\in A \\ v\in A}} W_e = \sum_{u\in A} \sum_{\substack{e=(u,v)\in E \\ v\in A}} W_e \le \sum_{u\in A} \sum_{\substack{e=(u,v)\in E \\ v\in B}} W_e = W(A,B)$$

$$2\sum_{\substack{(u,v)\in E \\ u\in B \\ v\in B}} W_e \le W(A,B)$$

$$\sum_{e\in E} W_e = \sum_{\substack{e=(u,v) \\ u\in A \\ v\in A}} W_e + \sum_{\substack{e=(u,v) \\ u\in B \\ v\in B}} W_e + \sum_{e\in\delta(A,B)} W_e$$

$$\le \frac{1}{2}W(A,B) + \frac{1}{2}W(A,B) + W(A,B)$$

$$\therefore \quad W(A,B) \ge \left(\sum_{e\in E} W_e\right)/2 \ge \frac{OPT}{2}$$

Accelerate.

Idea: update only when there is a big improvement.

flip a node u only when it increases $w(A,B)$ by a fraction of at least $\frac{\varepsilon}{|V|}$ $\quad |V| = n$

$$w(A',B') \geqslant \left(1+\frac{\varepsilon}{n}\right) w(A,B)$$

$$\left(1+\frac{\varepsilon}{n}\right)^{\frac{n}{\varepsilon}} \geqslant 2.$$ (~~o. determined~~ omitted)

$O\left(\frac{n}{\varepsilon} \log W\right)$ iterations