Last class

- Multi-objective optimization
- NSGA-II
- SMS-EMOA
- MOEA/D

} Popular variants of MOEA

Heuristic Search and Evolutionary Algorithms

Lecture 12: Evolutionary Algorithms for Constrained Optimization

Chao Qian （钱超）

Associate Professor, Nanjing University, China

Email: qianc@nju.edu.cn
Homepage: http://www.lamda.nju.edu.cn/qianc/

Constrained optimization

General formulation:

$$\min_{x \in \mathcal{X}} \quad f(x)$$

$$s.t. \quad g_i(x) = 0, \quad 1 \le i \le q;$$

$$h_i(x) \le 0, \quad q+1 \le i \le m$$

objective function

equality constraints

inequality constraints

A solution is **(in)feasible** if it does (not) satisfy the constraints

The goal: find a feasible solution minimizing the objective f

Example – knapsack

Knapsack problem: given n items, each with a weight w_i and a value v_i, to select a subset of items maximizing the sum of values while keeping the summed weights within some capacity W_{max}

$$\arg max_{\boldsymbol{x}\in\{0,1\}^n} \sum_{i=1}^{n} v_i x_i \quad s.t. \sum_{i=1}^{n} w_i x_i \le W_{max}$$

$x_i = 1$: the i-th item is included

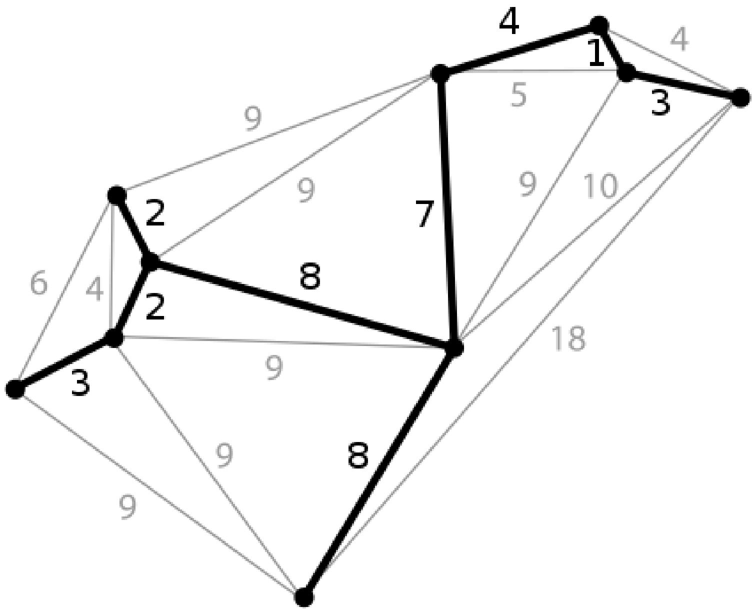
objective function

constraint

Example – minimum spanning tree

Minimum spanning tree problem:

given an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive weights $w: E \to \mathrm{R}^+$, to find a connected subgraph $E' \subseteq E$ with the minimum weight



$$\arg min_{\boldsymbol{x} \in \{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t. \quad c(\boldsymbol{x}) = 1$$

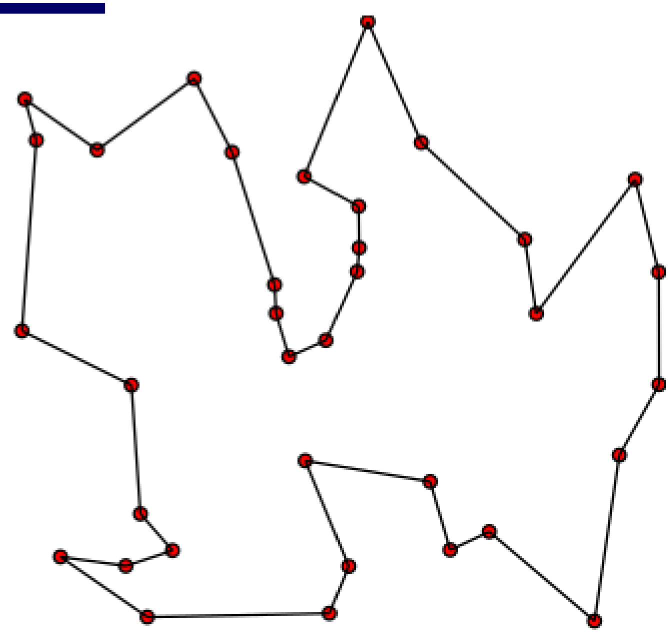
objective function

$x_i = 1$: the i-th edge is selected

constraint

$c(\boldsymbol{x})$: the number of connected components

Example – traveling salesman



Traveling salesman problem:

given an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive weights $w: E \to \mathrm{R}^+$, to find a Hamilton cycle with the minimum weight

$$\arg min_x \ w(x) \quad s.t. \ x \text{ is a Hamilton cycle}$$

Objective function: the sum of the edge weights on the cycle

Constraint: visit each vertex exactly once, starting and ending in the same vertex

EAs for constrained optimization

How to deal with constraints when EAs are used for constrained optimization?

The optimization problems in real-world applications often come with constraints

Constraint handling strategies

The final output solution must satisfy the constraints

Common constraint handling strategies

- Penalty functions
- Repair functions
- Restricting search to the feasible region
- Decoder functions

Penalty functions

Penalty functions: add penalties on the fitness of infeasible solutions

constrained

$$\begin{aligned} & min \quad f(x) \\ & s.t. \quad g_i(x) = 0, \quad 1 \le i \le q; \\ & \qquad\; h_i(x) \le 0, \quad q+1 \le i \le m \end{aligned}$$

unconstrained

$$min \quad f(x) + \sum_{i=1}^{m} \lambda_i \cdot f_i(x)$$

the i-th constraint violation degree

$$f_i(x) = \begin{cases} |g_i(x)| & 1 \le i \le q \\ \max\{0, h_i(x)\} & q+1 \le i \le m \end{cases}$$

Penalty functions

Penalty functions: add penalties on the fitness of infeasible solutions

constrained → unconstrained

$$\begin{aligned} &\min \quad f(x) \\ &s.t. \quad g_i(x) = 0, \quad 1 \le i \le q; \\ &\qquad\;\; h_i(x) \le 0, \quad q+1 \le i \le m \end{aligned}$$

$$\min \quad f(x) + \sum_{i=1}^{m} \lambda_i \cdot f_i(x)$$

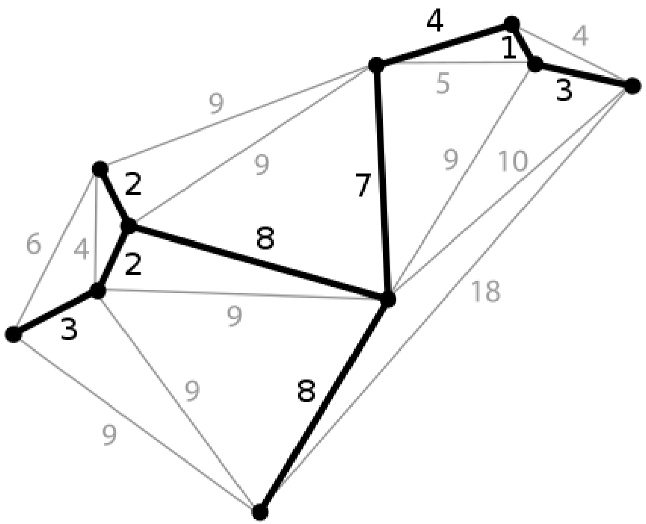
Requirement: the optimal solutions of the original and transformed problems should be consistent

- e.g., all λ_i are equal, and large enough: compare the constraint violation degrees first; if they are the same, compare the objective values f

Penalty functions

Minimum spanning tree problem:

given an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive weights $w: E \to \mathrm{R}^+$, to find a connected subgraph $E' \subseteq E$ with the minimum weight



$$\arg min_{\boldsymbol{x} \in \{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t.\ c(\boldsymbol{x}) = 1$$

Fitness function:

$$min \quad (c(\boldsymbol{x}) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$$

Constraint violation degree

$n^2 \cdot w_{max}$

Original objective function

Repair functions

Repair functions: repair infeasible solutions to feasible

Example - Knapsack: given n items, each with a weight w_i and a value v_i, to select a subset of items maximizing the sum of values while keeping the summed weights within some capacity W_{max}

$$\arg max_{x\in\{0,1\}^n} \sum_{i=1}^{n} v_i x_i \quad s.t. \sum_{i=1}^{n} w_i x_i \le W_{max}$$

v_i: 4,2,6,10,4,3,7,2; w_i: 2,3,3,8,6,5,7,1; $W_{max} = 25$

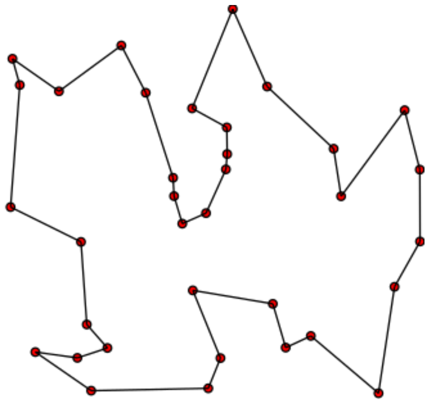
infeasible	1	1	0	1	1	0	1	1
feasible	1	1	0	1	1	0	0	1

Repairing: scan from left to right, and keep the value 1 if the summed weight does not exceed W_{max}

Restricting search to the feasible region

Restricting search to the feasible region: preserving feasibility by special initialization and reproduction

Example - traveling salesman: given an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive weights $w: E \to \mathrm{R}^+$, to find a Hamilton cycle with the minimum weight



$$\arg min_x \, w(x) \quad s.t. \; x \text{ is a Hamilton cycle}$$

Integer vector representation: the order of visiting vertexes

Permutation is feasible

1	6	2	5	7	4	8	3

Initialize with permutation; Apply mutation and recombination operators for permutation representation

Decoder functions

Decoder functions: map each genotype to a feasible phenotype

Example - Knapsack: given n items, each with a weight w_i and a value v_i, to select a subset of items maximizing the sum of values while keeping the summed weights within some capacity W_{max}

$$\arg max_{x\in\{0,1\}^n} \sum_{i=1}^{n} v_i x_i \quad s.t. \sum_{i=1}^{n} w_i x_i \leq W_{max}$$

v_i: 4,2,6,10,4,3,7,2; w_i: 2,3,3,8,6,5,7,1; $W_{max} = 25$

genotype	1	1	0	1	1	0	1	1
phenotype	1	1	0	1	1	0	0	1

Decoding: scan from left to right, and keep the value 1 if the summed weight does not exceed W_{max}

Constraint handling strategies

The final output solution must satisfy the constraints

⇩

Common constraint handling strategies

- Penalty functions
- Repair functions
- Restricting search to the feasible region
- Decoder functions

Other effective constraint handling strategies?

(1+1)-EA for MST

Minimum spanning tree (MST):

- Given: an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive integer weights $w: E \to \mathbb{N}^+$
- The Goal: find a connected subgraph $E' \subseteq E$ with the minimum weight

Formulation: $\arg min_{\boldsymbol{x} \in \{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t. \;\; c(\boldsymbol{x}) = 1$

$\boldsymbol{x} \in \{0,1\}^m \leftrightarrow$ a subgraph

$x_i = 1$ means that edge e_i is selected

(1+1)-EA for MST

(1+1)-EA: Given a pseudo-Boolean function f:

1. $\boldsymbol{x} :=$ randomly selected from $\{0,1\}^n$.
2. Repeat until some termination criterion is met
3. $\quad \boldsymbol{x}' :=$ flip each bit of $\boldsymbol{x}$ with probability $1/n$.
4. $\quad$ if $f(\boldsymbol{x}') \leq f(\boldsymbol{x})$
5. $\quad\quad \boldsymbol{x} = \boldsymbol{x}'$.

Using the strategy of penalty functions

Fitness function:

$$min \ (c(\boldsymbol{x}) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$$

- $(c(\boldsymbol{x}) - 1)$: Constraint violation degree
- w_{ub}: $n^2 \cdot w_{max}$
- $\sum_{i:x_i=1} w_i$: Original objective function

Theorem. [Neumann & Wegener, TCS'07; Doerr et al., Algorithmica'12] The expected running time of the (1+1)-EA solving the MST problem is $O(m^2(\log n + \log w_{max}))$.

MST by MOEAs

$$\arg min_{\boldsymbol{x}\in\{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t. \; c(\boldsymbol{x}) = 1$$

⇩

Bi-objective reformulation $min \; (c(\boldsymbol{x}), \sum_{i:x_i=1} w_i)$

Theorem. [Neumann & Wegener, Nature Computing'05] The expected running time of the GSEMO solving the MST problem is $O(mn\,(n+\log w_{max}))$.

Penalty functions: $O(m^2(\log n + \log w_{max}))$

Bi-objective reformulation: $O(mn(n+\log w_{max}))$

Bi-objective reformulation is better for dense graphs, e.g., $m \in \Theta(n^2)$

MST by MOEAs

$$\arg\min_{\boldsymbol{x}\in\{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t.\ \ c(\boldsymbol{x}) = 1$$

⇩

Bi-objective reformulation $\min \ \left(c(\boldsymbol{x}), \sum_{i:x_i=1} w_i\right)$

GSEMO: Given a pseudo-Boolean function vector $\boldsymbol{f}$:

1. $\boldsymbol{x} :=$ randomly selected from $\{0,1\}^n$.
2. $P := \{\boldsymbol{x}\}$.
3. Repeat until some termination criterion is met
4. Choose $\boldsymbol{x}$ from P uniformly at random.
5. $\boldsymbol{x}' :=$ flip each bit of $\boldsymbol{x}$ with probability $1/n$.
6. if $\nexists\, \boldsymbol{z} \in P$ such that $\boldsymbol{z} \prec \boldsymbol{x}'$
7. $P := (P - \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \preccurlyeq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$.

Keep the non-dominated solutions generated so-far

Proof

Main idea:

(1) obtain the empty subgraph 0^n

(2) obtain a minimum spanning tree

The analysis of phase (1): $min \ \ (c(\boldsymbol{x}), w(\boldsymbol{x}) = \sum_{i:x_i=1} w_i)$

Using multiplicative drift analysis:

- design the distance function: $V(P) = min\,\{w(\boldsymbol{x}) \mid \boldsymbol{x} \in P\}$ — the minimum weight in the population
- analyze the expected drift:

$$\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = P] \geq \frac{1}{n} \cdot \frac{1}{m}(1-\frac{1}{m})^{m-1} \cdot \sum_{i=1}^{|\boldsymbol{x}^*|}\big(w(\boldsymbol{x}^*) - w(\boldsymbol{y}^i)\big)$$

$\boldsymbol{y}^i$: the resulting solution

select the solution $\boldsymbol{x}^*$ with the smallest $w(\boldsymbol{x})$ value, and flip only one 1-bit

Proof

Main idea:

(1) obtain the empty subgraph 0^n

(2) obtain a minimum spanning tree

The analysis of phase (1): $min \ \ (c(\boldsymbol{x}), w(\boldsymbol{x}) = \sum_{i:x_i=1} w_i)$

Using multiplicative drift analysis:

- design the distance function: $V(P) = min\ \{w(\boldsymbol{x}) \mid \boldsymbol{x} \in P\}$
- analyze the expected drift:

$$\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = P] \geq \frac{1}{n} \cdot \frac{1}{m}(1 - \frac{1}{m})^{m-1} \cdot \sum_{i=1}^{|\boldsymbol{x}^*|} \big(w(\boldsymbol{x}^*) - w(\boldsymbol{y}^i)\big)$$

$$= \frac{1}{n} \cdot \frac{1}{m}(1 - \frac{1}{m})^{m-1} \cdot w(\boldsymbol{x}^*)$$

$$= \frac{1}{n} \cdot \frac{1}{m}(1 - \frac{1}{m})^{m-1} \cdot V(\xi_t)$$

Proof

Main idea:

(1) obtain the empty subgraph 0^n

(2) obtain a minimum spanning tree

The analysis of phase (1): $min \ \ (c(\boldsymbol{x}),\ w(\boldsymbol{x}) = \sum_{i:x_i=1} w_i)$

Using multiplicative drift analysis:

- design the distance function: $V(P) = min\,\{w(\boldsymbol{x}) \mid \boldsymbol{x} \in P\}$
- analyze the expected drift: $\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = P] \geq \frac{1}{emn} \cdot V(\xi_t)$

Upper bound on the expected running time:

$$\sum_P \pi_0(P) \cdot \frac{1 + \ln\,(V(P)/V_{min})}{\delta} \leq emn(1 + \ln\,(mw_{max}))$$

$V(P) \leq mw_{max}$ $\quad$ $V_{min} \geq 1$

Proof

Main idea:

(1) obtain the empty subgraph 0^n

(2) obtain a minimum spanning tree

The analysis of phase (1): $min \ \ (c(\boldsymbol{x}), w(\boldsymbol{x}) = \sum_{i:x_i=1} w_i)$

Using multiplicative drift analysis:

- design the distance function: $V(P) = min\,\{w(\boldsymbol{x}) \mid \boldsymbol{x} \in P\}$
- analyze the expected drift: $\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = P] \geq \frac{1}{emn} \cdot V(\xi_t)$

Upper bound on the expected running time:

$$\sum_P \pi_0(P) \cdot \frac{1 + \ln\,(V(P)/V_{min})}{\delta} \leq emn(1 + \ln\,(mw_{max}))$$

$$\in O(mn\,(\log n + \log w_{max}))$$

Proof

The analysis of phase (2): $\min \ (c(\boldsymbol{x}),\ w(\boldsymbol{x}) = \sum_{i:x_i=1} w_i)$

$\boldsymbol{x}^i$: the Pareto optimal solution with i connected components

- the found Pareto optimal solutions will always be kept
- follow the path: $\boldsymbol{x}^n \rightarrow \boldsymbol{x}^{n-1} \rightarrow \cdots \rightarrow \boldsymbol{x}^2 \rightarrow \boldsymbol{x}^1 \rightarrow$ a minimum spanning tree

0^n

the probability is at least : $\frac{1}{n} \cdot \frac{1}{m}(1-\frac{1}{m})^{m-1} \geq \frac{1}{emn}$

The expected running time is at most: $(n-1) \cdot emn \in O(mn^2)$

The expected running time of phase (1): $O(mn(\log n + \log w_{max}))$

The total expected running time: $O(mn(n + \log w_{max}))$

MST by MOEAs

$$\arg min_{\boldsymbol{x}\in\{0,1\}^m} \sum_{i=1}^{m} w_i x_i \quad s.t.\ \ c(\boldsymbol{x}) = 1$$

Bi-objective reformulation $min\ \ (c(\boldsymbol{x}), \sum_{i:x_i=1} w_i)$

Theorem. [Neumann & Wegener, Nature Computing'05] The expected running time of the GSEMO solving the MST problem is $O(mn\ (n + \log w_{max}))$.

Penalty functions: $O(m^2(\log n + \log w_{max}))$

Bi-objective reformulation: $O(mn(n + \log w_{max}))$

Bi-objective reformulation is better for dense graphs, e.g., $m \in \Theta(n^2)$

More examples

Problem	Penalty functions	Bi-objective reformulation
Set cover	*exponential*	$O(mn(\log c_{max} + \log n))$ [Friedrich et al., ECJ'10]
Minimum cut	*exponential*	$O(Fm(\log c_{max} + \log n))$ [Neumann et al., Algorithmica'11]
Minimum label spanning tree	$\Omega(ku^k)$	$O(k^2 \log k)$ [Lai et al., TEC'14]
Minimum cost coverage	*exponential*	$O(Nn(\log n + \log w_{max} + N))$ [Qian et al., IJCAI'15]

Better

Bi-objective reformulation

Main idea:

1. transform the original constrained optimization problem into a bi-objective optimization problem

constrained

$$\begin{aligned} &min \;\; f(x) \\ &s.t. \;\; g_i(x) = 0, \quad 1 \le i \le q; \\ &\qquad\;\; h_i(x) \le 0, \quad q+1 \le i \le m \end{aligned}$$

bi-objective

$$min \;\; \left(f(x),\ \textstyle\sum_{i=1}^{m} f_i(x)\right)$$

constraint violation degree

$$f_i(x) = \begin{cases} |g_i(x)| & 1 \le i \le q \\ \max\{0, h_i(x)\} & q+1 \le i \le m \end{cases}$$

Bi-objective reformulation

Main idea:

1. transform the original constrained optimization problem into a bi-objective optimization problem

constrained

$$\begin{aligned} &min \quad f(x) \\ &s.t. \quad g_i(x) = 0, \quad 1 \le i \le q; \\ &\qquad\; h_i(x) \le 0, \quad q+1 \le i \le m \end{aligned}$$

bi-objective

$$min \quad (f(x), \ \sum_{i=1}^{m} f_i(x))$$

2. employ a multi-objective EA to solve the transformed problem

constraint violation degree = 0

3. output the feasible solution from the generated non-dominated solution set

Constraint handling strategies

The final output solution must satisfy the constraints

⇩

Common constraint handling strategies

- Penalty functions
- Repair functions
- Restricting search to the feasible region
- Decoder functions
- Bi-objective reformulation

Search only in the feasible region (Repair functions, Restricting search to the feasible region, Decoder functions)

allow infeasible solutions in the search

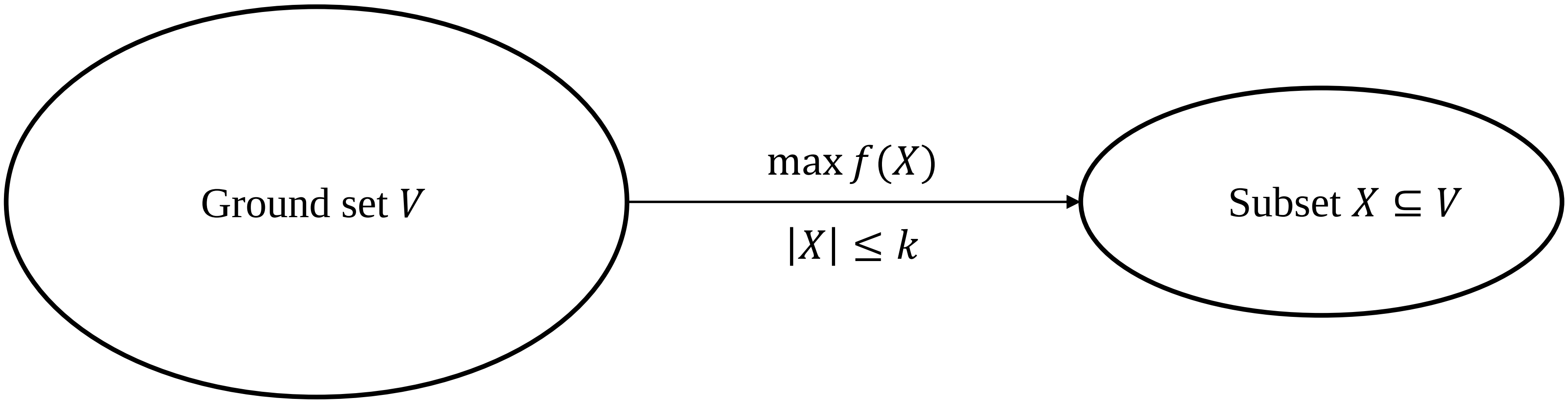
↪ Better algorithms?

Subset selection

Subset selection is to select a subset of size k from a total set of n items for optimizing some objective function

Formally stated: given all items $V = \{v_1, \ldots, v_n\}$, an objective function $f: 2^V \to \mathrm{R}$ and a budget k, to find a subset $X \subseteq V$ such that

$$max_{X \subseteq V} \quad f(X) \quad s.t. \quad |X| \leq k.$$



Application - sparse regression

Sparse regression [Tropp, TIT'04] : select a few observation variables to best approximate the predictor variable by linear regression

observation variables

predictor variable z

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.03	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.93	0.56

a subset X of observation variables

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.03	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.93	0.56

Item v_i: an observation variable

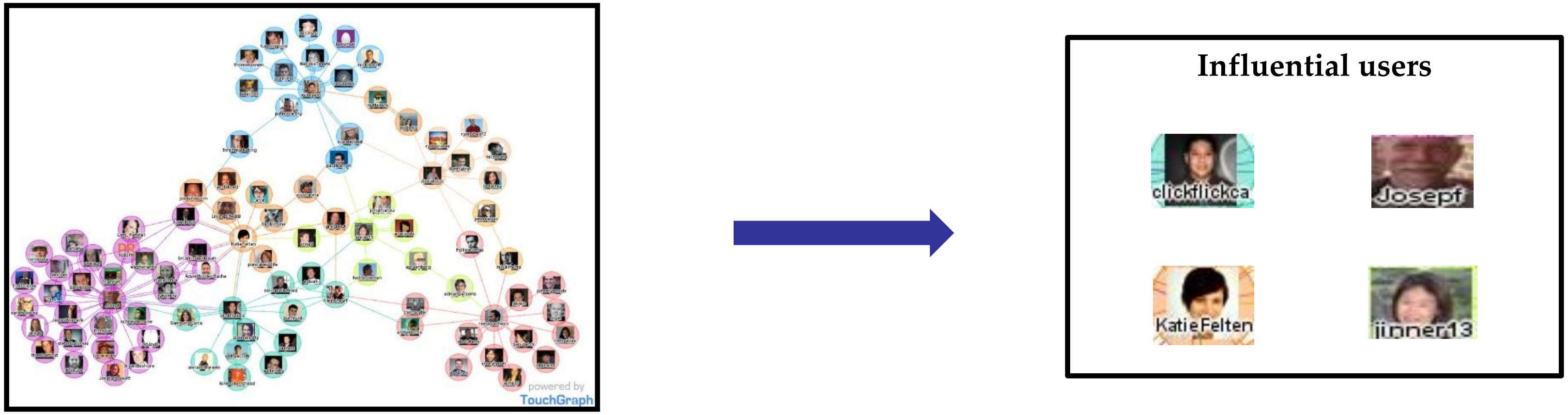
variance

mean squared error

Objective f: squared multiple correlation $R^2_{z,X} = \dfrac{\mathrm{Var}(z) - \mathrm{MSE}_{z,X}}{\mathrm{Var}(z)}$

Application - influence maximization

Influence maximization [Kempe et al., KDD'03] : select a subset of users from a social network to maximize its influence spread

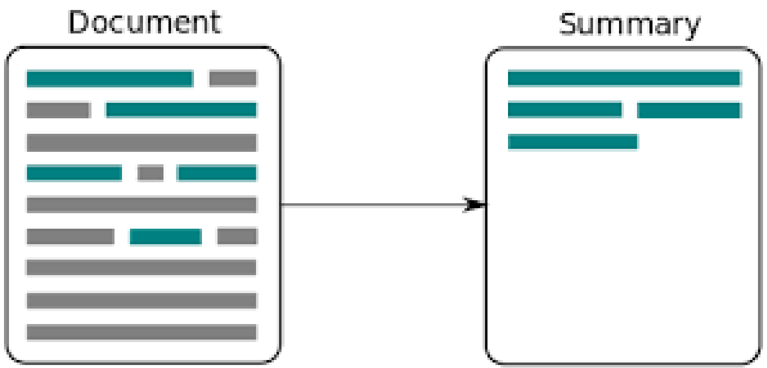


Item v_i: a social network user

Objective f: influence spread, measured by the expected number of social network users activated by diffusion

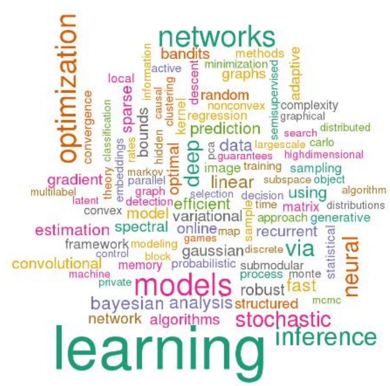
Application - document summarization

Document summarization [Lin & Bilmes, ACL'11] : select a few sentences to best summarize the documents



Item v_i: a sentence

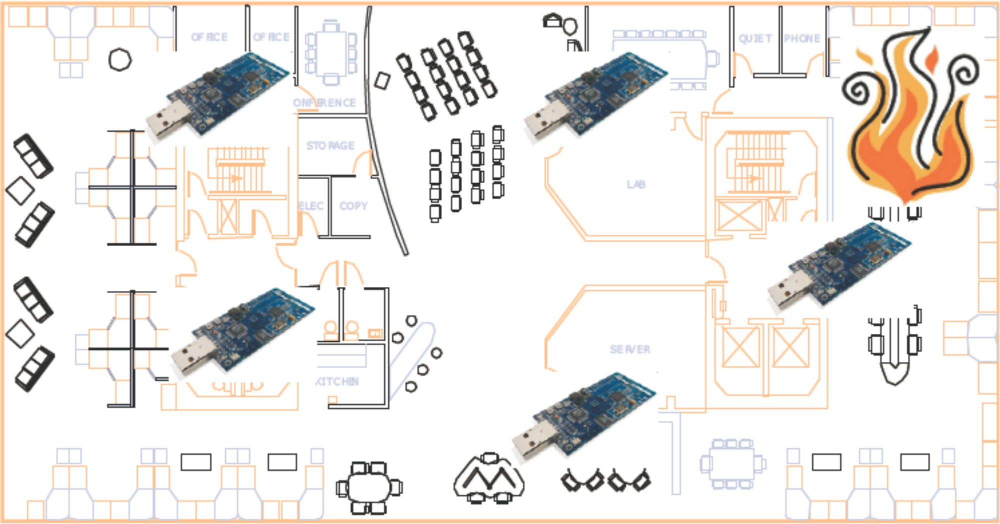
Objective f: summary quality



Application - sensor placement

Sensor placement [Krause & Guestrin, IJCAI'09 Tutorial] : select a few places to install sensors such that the information gathered is maximized

Water contamination detection



Fire detection

Item v_i: a place to install a sensor

Objective f: entropy

Subset selection

Sparse regression

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.68	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.08	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.98	0.56

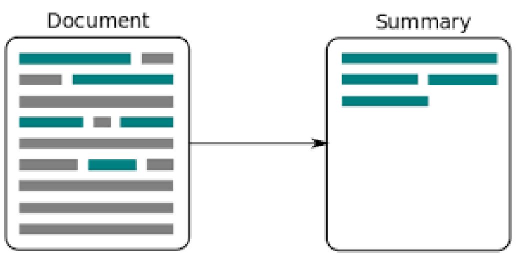
Machine learning

Data mining

Influence maximization

Subset selection

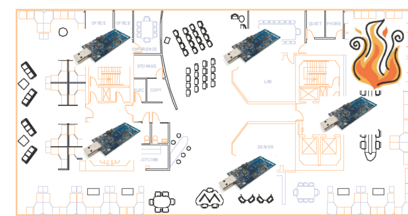
Document summarization

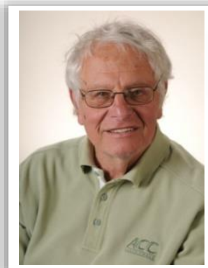


Natural language processing

Networks

Sensor placement





George Nemhauser

John Von Neumann Theory Prize

[Mathematical Programming 1978]

f : monotone and submodular

The greedy algorithm:

$(1 - 1/e)$-approximation

Best Paper/Test of Time Award:

[Kempe et al., KDD'03]

[Das & Kempe, ICML'11]

[Iyer & Bilmes, NIPS'13]

Subset representation

A subset $X \subseteq V$ can be naturally represented by a Boolean vector $\boldsymbol{x} \in \{0,1\}^n$

- the i-th bit $x_i = 1$ if the item $v_i \in X$; $x_i = 0$ otherwise
- $X = \{v_i \mid x_i = 1\}$

$V = \{v_1, v_2, v_3, v_4, v_5\}$

a subset $X \subseteq V$		a Boolean vector $\boldsymbol{x} \in \{0,1\}^5$
$\emptyset$		00000
$\{v_1\}$	$\Leftrightarrow$	10000
$\{v_2, v_3, v_5\}$		01101
$\{v_1, v_2, v_3, v_4, v_5\}$		11111

POSS algorithm

POSS algorithm [Qian, Yu and Zhou, NIPS'15]

Transformation:

$$max_{X \subseteq V} \; f(X) \quad s.t. \quad |X| \leq k$$ original

$$\Downarrow$$

$$min_{X \subseteq V} \; (-f(X), |X|)$$ bi-objective

Algorithm 1 POSS

Input: all variables $V = \{X_1, \ldots, X_n\}$, a given objective f and an integer parameter $k \in [1, n]$
Parameter: the number of iterations T
Output: a subset of V with at most k variables
Process:

```
Let s = {0}^n and P = {s}.
Let t = 0.
while t < T do
    Select s from P uniformly at random.
    Generate s' by flipping each bit of s with prob. 1/n.
    Evaluate f_1(s') and f_2(s').
    if ∄z ∈ P such that z ≺ s' then
        Q = {z ∈ P | s' ⪯ z}.
        P = (P \ Q) ∪ {s'}.
    end if
    t = t + 1.
end while
return arg min_{s∈P,|s|≤k} f_1(s)
```

Initialization: put the special solution $\{0\}^n$ into the population P

Parent selection & Reproduction: pick a solution $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x}$ with prob. $1/n$ to generate a new solution

Evaluation & Survivor selection: if the new solution is not dominated, put it into P and weed out bad solutions

Output: select the best feasible solution

Sparse regression

Sparse regression: given all observation variables $V = \{v_1, \dots, v_n\}$, a predictor variable z and a budget k, to find a subset $X \subseteq V$ such that

$$max_{X \subseteq V} \quad R^2_{z,X} = \frac{\mathrm{Var}(z) - \mathrm{MSE}_{z,X}}{\mathrm{Var}(z)} \quad s.t. \quad |X| \leq k$$

$\mathrm{Var}(z)$: variance of z

$\mathrm{MSE}_{z,X}$: mean squared error of predicting z by using observation variables in X

observation variables

predictor variable z

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.03	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.93	0.56

a subset X of observation variables

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.03	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.93	0.56

Experimental results

the size constraint: $k = 8$ **the number of iterations of POSS: $2ek^2n$**

exhaustive search — OPT; greedy algorithms — FR, FoBa, OMP, RFE; relaxation methods — MCP

Data set	OPT	POSS	FR	FoBa	OMP	RFE	MCP
housing	.7437±.0297	.7437±.0297	.7429±.0300•	.7423±.0301•	.7415±.0300•	.7388±.0304•	.7354±.0297•
eunite2001	.8484±.0132	.8482±.0132	.8348±.0143•	.8442±.0144•	.8349±.0150•	.8424±.0153•	.8320±.0150•
svmguide3	.2705±.0255	.2701±.0257	.2615±.0260•	.2601±.0279•	.2557±.0270•	.2136±.0325•	.2397±.0237•
ionosphere	.5995±.0326	.5990±.0329	.5920±.0352•	.5929±.0346•	.5921±.0353•	.5832±.0415•	.5740±.0348•
sonar	–	.5365±.0410	.5171±.0440•	.5138±.0432•	.5112±.0425•	.4321±.0636•	.4496±.0482•
triazines	–	.4301±.0603	.4150±.0592•	.4107±.0600•	.4073±.0591•	.3615±.0712•	.3793±.0584•
coil2000	–	.0627±.0076	.0624±.0076•	.0619±.0075•	.0619±.0075•	.0363±.0141•	.0570±.0075•
mushrooms	–	.9912±.0020	.9909±.0021•	.9909±.0022•	.9909±.0022•	.6813±.1294•	.8652±.0474•
clean1	–	.4368±.0300	.4169±.0299•	.4145±.0309•	.4132±.0315•	.1596±.0562•	.3563±.0364•
w5a	–	.3376±.0267	.3319±.0247•	.3341±.0258•	.3313±.0246•	.3342±.0276•	.2694±.0385•
gisette	–	.7265±.0098	.7001±.0116•	.6747±.0145•	.6731±.0134•	.5360±.0318•	.5709±.0123•
farm-ads	–	.4217±.0100	.4196±.0101•	.4170±.0113•	.4170±.0113•	–	.3771±.0110•
POSS: win/tie/loss		–	12/0/0	12/0/0	12/0/0	11/0/0	12/0/0

• denotes that POSS is significantly better by the t-test with confidence level 0.05

POSS is significantly better than all the compared state-of-the art algorithms on all data sets

Theoretical analysis

POSS can achieve the optimal polynomial-time approximation guarantee

Theorem 1. For subset selection with monotone objective functions, POSS using $E[T] \leq 2ek^2n$ finds a solution X with $|X| \leq k$ and

$$f(X) \geq \boxed{(1 - e^{-\gamma})} \cdot \mathrm{OPT}.$$

the optimal polynomial-time approximation guarantee for monotone f [Harshaw et al., ICML'19]

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

submodularity ratio [Das & Kempe, ICML'11]

the optimal function value

Roughly speaking, the improvement by adding a specific item is proportional to the current distance to the optimum

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^i\right) \cdot \mathrm{OPT}$

$i = 0$

initial solution $00 \ldots 0$

$|00 \ldots 0| = 0$

$f(00 \ldots 0) = 0$

$i = k$

$$1 - \left(1 - \frac{\gamma}{k}\right)^k = 1 - \left(1 - \frac{1}{k/\gamma}\right)^{(k/\gamma)\cdot\gamma}$$

let $m = k/\gamma$

$$\geq 1 - e^{-\gamma}$$

$(1 - 1/m)^m \leq 1/e$

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^i\right) \cdot \mathrm{OPT}$

$i = 0$ → ? → $i = k$

initial solution $00 \ldots 0$

$|00 \ldots 0| = 0$

$f(00 \ldots 0) = 0$

$$1 - \left(1 - \frac{\gamma}{k}\right)^k \geq 1 - e^{-\gamma}$$

the desired approximation guarantee

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^i\right) \cdot \mathrm{OPT}$
- in each iteration of POSS:
  - select $\boldsymbol{x}$ from the population P
  - flip one specific 0-bit of $\boldsymbol{x}$ to 1-bit (i.e., add the specific item $\hat{v}$ in Lemma 1)

$$|\boldsymbol{x}'| = |\boldsymbol{x}| + 1 \leq i + 1 \text{ and } f(\boldsymbol{x}') \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^{i+1}\right) \cdot \mathrm{OPT}$$

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

$$f(\boldsymbol{x}') - f(\boldsymbol{x}) \geq \frac{\gamma}{k} \cdot \left(\mathrm{OPT} - f(\boldsymbol{x})\right)$$

$$f(\boldsymbol{x}') \geq \left(1 - \frac{\gamma}{k}\right) f(\boldsymbol{x}) + \frac{\gamma}{k} \cdot \mathrm{OPT}$$

$$f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^{i}\right) \cdot \mathrm{OPT}$$

$$f(\boldsymbol{x}') \geq \left(1 - \frac{\gamma}{k}\right)\left(1 - \left(1 - \frac{\gamma}{k}\right)^{i}\right) \cdot \mathrm{OPT} + \frac{\gamma}{k} \cdot \mathrm{OPT} = \left(1 - \left(1 - \frac{\gamma}{k}\right)^{i+1}\right) \cdot \mathrm{OPT}$$

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^{i}\right) \cdot \mathrm{OPT}$
- in each iteration of POSS:
  - select $\boldsymbol{x}$ from the population P, the probability: $\frac{1}{|P|}$
  - flip one specific 0-bit of $\boldsymbol{x}$ to 1-bit, the probability: $\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{en}$

    (i.e., add the specific item $\hat{v}$ in Lemma 1)

$$|\boldsymbol{x}'| = |\boldsymbol{x}| + 1 \leq i + 1 \text{ and } f(\boldsymbol{x}') \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^{i+1}\right) \cdot \mathrm{OPT}$$

$i \longrightarrow i + 1$ the probability: $\frac{1}{|P|} \cdot \frac{1}{en}$

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^i\right) \cdot \mathrm{OPT}$
- in each iteration of POSS:

$i \longrightarrow i+1$ the probability: $\frac{1}{|P|} \cdot \frac{1}{en}$ $\xrightarrow{|P| \leq 2k}$ $\frac{1}{2ekn}$

➢ Exclude solutions with size at least $2k$

➢ The solutions in P are always incomparable

→ For each size in $\{0,1,\ldots,2k-1\}$, there exists at most one solution in P

Proof

Lemma 1. For any $X \subseteq V$, there exists one item $\hat{v} \in V \setminus X$ such that

$$f(X \cup \{\hat{v}\}) - f(X) \geq \frac{\gamma}{k}(\mathrm{OPT} - f(X))$$

a subset

Main idea:

- consider a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq i$ and $f(\boldsymbol{x}) \geq \left(1 - \left(1 - \frac{\gamma}{k}\right)^i\right) \cdot \mathrm{OPT}$
- in each iteration of POSS:

$i \longrightarrow i+1$ the probability: $\frac{1}{|P|} \cdot \frac{1}{en} \xrightarrow{|P| \leq 2k} \frac{1}{2ekn}$

$i \longrightarrow i+1$ the expected number of iterations: $2ekn$

$i = 0 \longrightarrow k$ the expected number of iterations: $k \cdot 2ekn$

Theoretical analysis:

The advantage of bi-objective reformulation for handling constraints

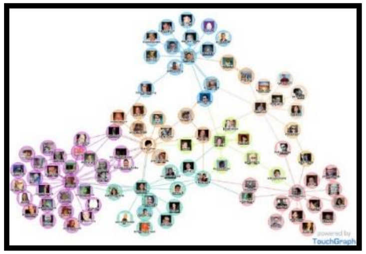
Algorithm design:

The POSS algorithm for subset selection

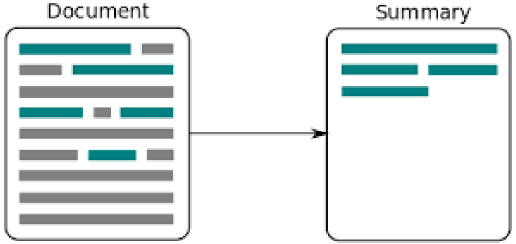
Sparse regression

	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
x8	0.1	0.03	0.32	···	···	0.33	0.51	0.44
x9	0.32	0.1	0.2	···	···	0.06	0.66	0
x10	0.24	0	0.02	···	···	0.6	0.03	0.33
x11	0.12	0.45	0.44	···	···	0.64	0.45	1
x12	0.36	0.58	0.12	···	···	0.73	0.58	0.67
x13	0.2	0.02	0.24	···	···	0.34	0.02	0.89
x14	0.24	0.92	0.33	···	···	0.24	0.93	0.56

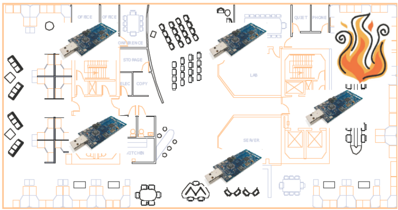
Influence maximization



Document summarization



Sensor placement



POSS algorithm

POSS algorithm [Qian, Yu and Zhou, NIPS'15]

Transformation:

$$\max_{X \subseteq V} \; f(X) \quad s.t. \quad |X| \le k \qquad \text{original}$$

⇩

$$\min_{X \subseteq V} \; (-f(X), |X|) \qquad \text{bi-objective}$$

Algorithm 1 POSS

Input: all variables $V = \{X_1, \ldots, X_n\}$, a given objective f and an integer parameter $k \in [1, n]$
Parameter: the number of iterations T
Output: a subset of V with at most k variables
Process:

Let $s = \{0\}^n$ and $P = \{s\}$.
Let $t = 0$.
while $t < T$ **do**
  Select s from P uniformly at random.
  Generate s' by flipping each bit of s with prob. $\frac{1}{n}$.
  Evaluate $f_1(s')$ and $f_2(s')$.
  if $\nexists z \in P$ such that $z \prec s'$ **then**
    $Q = \{z \in P \mid s' \preceq z\}$.
    $P = (P \setminus Q) \cup \{s'\}$.
  end if
  $t = t + 1$.
end while
return $\arg\min_{s \in P, |s| \le k} f_1(s)$

Parent selection & Reproduction: pick a solution $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x}$ with prob. $1/n$ to generate a new solution

Using bit-wise mutation only

PORSS algorithm

PORSS algorithm [Qian, Bian and Feng, AAAI'20]

Transformation:

$$\max_{X \subseteq V} \; f(X) \quad s.t. \quad |X| \le k$$ original

⇩

$$\min_{X \subseteq V} \; (-f(X), |X|)$$ bi-objective

Algorithm 2 PORSS Algorithm

Input: $V = \{v_1, \ldots, v_n\}$; objective $f : 2^V \to \mathbb{R}$; budget k
Parameter: the number T of iterations
Output: a subset of V with at most k items
Process:

```
Let x = 0^n, P = {x} and t = 0;
while t < T do
    Select x, y from P randomly with replacement;
    Apply recombination on x, y to generate x', y';
    Apply bit-wise mutation on x', y' to generate x'', y'';
    for each q ∈ {x'', y''}
        if ∄z ∈ P such that z ≺ q then
            P = (P \ {z ∈ P | q ⪯ z}) ∪ {q}
        end if
    end for
    t = t + 1
end while
return arg max_{x∈P,|x|≤k} f(x)
```

Parent selection & Reproduction:

- pick two solutions randomly from P
- apply recombination operator
- apply bit-wise mutation operator

PORSS algorithm

- One-point crossover

Parent1	1	0	1	1	1	0	0	0
Parent2	0	0	1	0	1	0	1	0

Offspring1	1	0	1	1	1	0	1	0
Offspring2	0	0	1	0	1	0	0	0

- Uniform crossover

Parent1	1	0	1	1	1	0	0	0
Parent2	0	0	1	0	1	0	1	0

Offspring1	1	0	1	1	1	0	1	0
Offspring2	0	0	1	0	1	0	0	0

PORSS algorithm

the size constraint: $k = 8$

State-of-the-art algorithms

PORSS using one-point crossover

PORSS using uniform crossover

Data set	(#inst, #feat)	OPT	Greedy	POSS	$PORSS_o$	$PORSS_u$
svmguide3	(1243, 22)	0.221	0.214	0.220±0.001	0.220±0.001	**0.221±0.001**
triazines	(186, 60)	0.328	0.316	0.327±0.000	**0.328±0.000**	**0.328±0.000**
clean1	(476, 166)	–	0.371	0.386±0.004	0.387±0.006	**0.393±0.005**
usps	(7291, 256)	–	0.562	0.570±0.003	**0.572±0.003**	**0.572±0.003**
scene	(1211, 294)	–	0.254	0.268±0.003	**0.272±0.002**	0.271±0.002
protein	(17766, 356)	–	0.132	0.132±0.000	**0.133±0.000**	**0.133±0.000**
colon-cancer	(62, 2000)	–	0.890	0.906±0.011	0.909±0.018	**0.911±0.014**
cifar10	(50000, 3072)	–	0.069	0.070±0.001	0.070±0.001	**0.071±0.001**
leukemia	(72, 7129)	–	0.947	0.966±0.009	0.968±0.006	**0.969±0.007**
smallNORB	(24300, 18432)	–	0.461	0.535±0.007	0.547±0.003	**0.550±0.002**
POSS: Count of direct win			**9.5**	–	**1**	**0**
Average rank			3.95	2.95	1.85	1.25

PORSS performs the best

Summary

- Constrained optimization
- Constraint handling strategies
  - Penalty functions
  - Repair functions
  - Restricting search to the feasible region
  - Decoder functions
  - Bi-objective reformulation → Give an example of algorithm design guided by theoretical analysis

References

- A. E. Eiben and J. E. Smith. Introduction to Evolutionary Computing. Chapter 13.
- T. Friedrich, J. He, N. Hebbinghaus, F. Neumann and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. *Evolutionary Computation*, 2010, 18(4): 617-633
- B. Doerr, D. Johannsen and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 2012, 64: 673-697
- X. Lai, Y. Zhou, J. He and J. Zhang. Performance analysis of evolutionary algorithms for the minimum label spanning tree problem. *IEEE Transactions on Evolutionary Computation*, 2014, 18(6): 860-872
- F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 2006, 5(3): 305-319

References

- F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 2007, 378(1): 32-40
- F. Neumann, J. Reichel and M. Skutella. Computing minimum cuts by randomized search heuristics. *Algorithmica*, 2011, 59(3): 323-342
- C. Qian, Y. Yu and Z.-H. Zhou. On constrained Boolean Pareto optimization. In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, 2015, pages 389-395, Buenos Aires, Argentina
- C. Qian, Y. Yu and Z.-H. Zhou. Subset selection by Pareto optimization. In: *Advances in Neural Information Processing Systems 28 (NIPS'15)*, 2015, pages 1765-1773, Montreal, Canada
- C. Qian, C. Bian and C. Feng. Subset selection by Pareto optimization with recombination. In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*, New York, NY, 2020, pp.2408-2415.