http://lamda.nju.edu.cn

Pareto Optimization for Subset Selection: Theory and Applications in Machine Learning

Yang Yu[1] and Chao Qian[2]

[1]LAMDA Group, Nanjing University, China
[2]University of Science and Technology of China

Email: yuy@nju.edu.cn, chaoqian@ustc.edu.cn

Outline

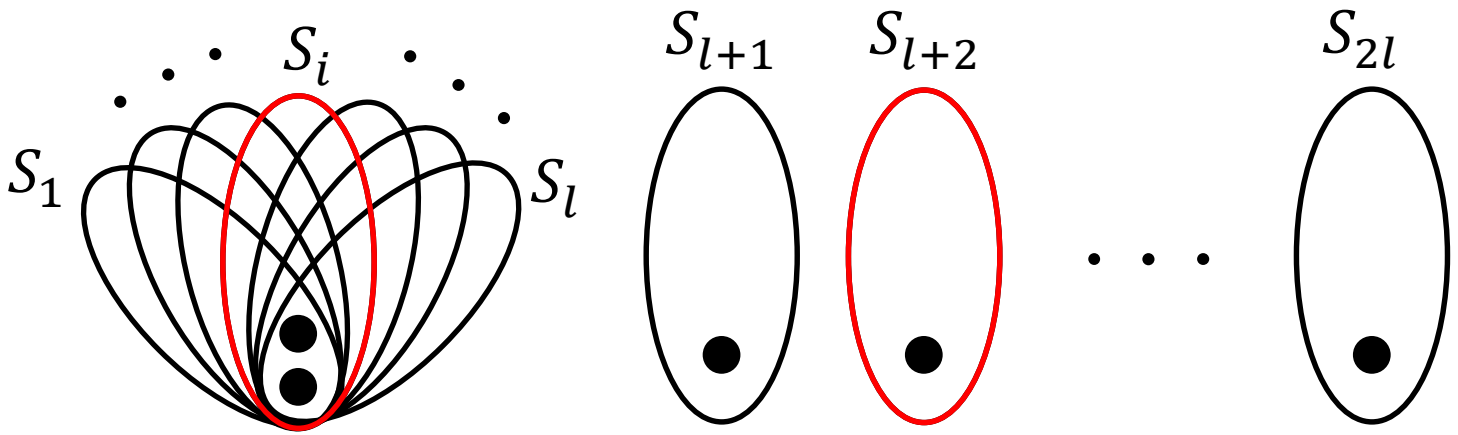
- **Introduction**
- Pareto optimization for subset selection
- Pareto optimization for large-scale subset selection
- Pareto optimization for noisy subset selection
- Conclusion

Maximum coverage

Maximum coverage [Feige, JACM'98]: select at most B sets from n given sets to make the union maximal

Formally stated: given a ground set U, a collection $V = \{S_1, \dots, S_n\}$ of subsets of U and a budget B, it is to find a subset $X \subseteq V$ such that

$$max_{X \subseteq V} \quad f(X) = |\cup_{S_i \in X} S_i| \quad s.t. \quad |X| \leq B.$$



Sparse regression

Sparse regression [Tropp, TIT'04]: find a sparse approximation solution to the linear regression problem

Formally stated: given all observation variables $V = \{v_1, \ldots, v_n\}$, a predictor variable z and a budget B, it is to find a subset $X \subseteq V$ such that

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

	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

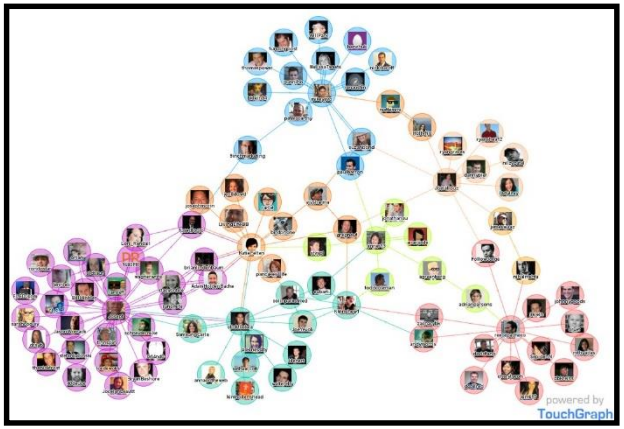
	Corr.	Dis.	LR	···	···	AIC.	BIC	RF.
x1	0.28	0.46	1	···	···	0.22	0.63	1
x2	0.31	0.59	0.64	···	···	0.58	0.56	1
x3	0.11	0.02	0.53	···	···	0.43	0.01	1
x4	0.1	0.1	0.64	···	···	0.73	0.92	1
x5	0.02	0.15	0.33	···	···	0.56	0.36	0.78
x6	0.36	0.02	0.01	···	···	0.32	0.02	0.22
x7	0.2	0.2	0.21	···	···	0.21	0.02	0.11
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

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

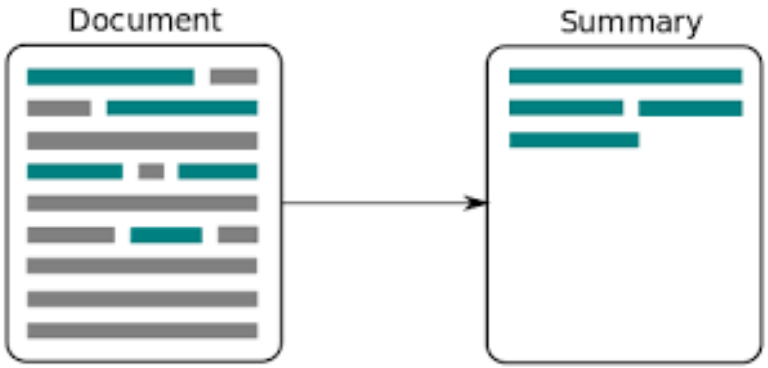
Formally stated: given a directed graph $G = (V, E)$ with $V = \{v_1, \dots, v_n\}$, edge probabilities $p_{u,v}$ $((u, v) \in E)$ and a budget B, it is to find a subset $X \subseteq V$ such that

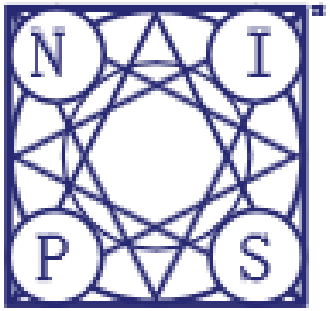
$$max_{X \subseteq V} \quad f(X) = \sum_{i=1}^{n} p(X \to v_i) \quad s.t. \quad |X| \leq B.$$

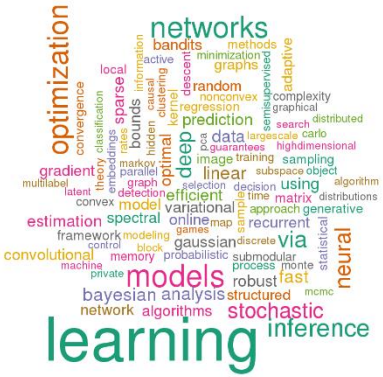


Document summarization

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



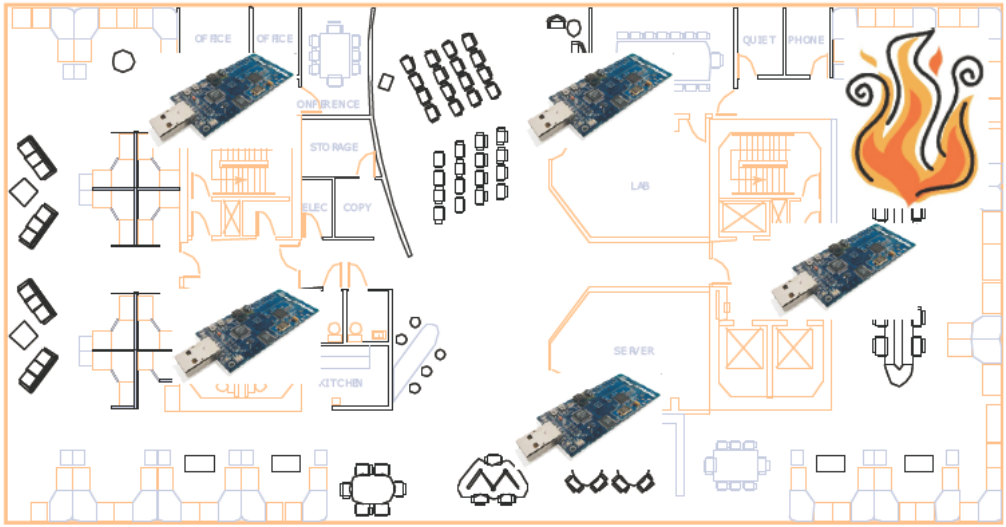




Sensor placement

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

Subset selection

Subset selection is to select a subset of size B 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 $f: 2^V \rightarrow \mathrm{R}$ and a budget B, it is to find a subset $X \subseteq V$ such that

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

Application	v_i	f
maximum coverage	a set of elements	size of the union
sparse reg		MSE of prediction
influence max		influence spread
document summarization	a sentence	summary quality
sensor placement	a place to install a sensor	entropy

Many applications, but NP-hard in general!

Subset selection - submodular

Subset selection: given all items $V = \{v_1, \ldots, v_n\}$, an objective function $f\colon 2^V \to \mathrm{R}$ and a budget B, it is to find a subset $X \subseteq V$ such that

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

Monotone: for any $X \subseteq Y \subseteq V$, $f(X) \leq f(Y)$

Submodular [Nemhauser et al., MP'78] : satisfy the natural diminishing returns property, i.e., for any $X \subseteq Y \subseteq V$, $v \notin Y$,

$$f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y);$$

or equivalently, for any $X \subseteq Y \subseteq V$,

Discrete analogue of convexity!

$$f(Y) - f(X) \leq \sum_{v \in Y \setminus \mathrm{X}} f(X \cup \{v\}) - f(X);$$

or equivalently, for any $X, Y \subseteq V$,

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y).$$

Subset selection – submodular examples

Maximum coverage: given a ground set U, a collection $V = \{S_1, \dots, S_n\}$ of subsets of U and a budget B, it is to find a subset $X \subseteq V$ such that

$$max_{X \subseteq V} \quad f(X) = |\cup_{S_i \in X} S_i| \quad s.t. \quad |X| \leq B.$$

Monotone: $\forall X \subseteq Y \subseteq V: f(X) \leq f(Y)$

Submodular: $\forall X \subseteq Y \subseteq V, v \notin Y: f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$

Subset selection – submodular ratio

Subset selection: given all items $V = \{v_1, \ldots, v_n\}$, an objective function $f\colon 2^V \to \mathrm{R}$ and a budget B, it is to find a subset $X \subseteq V$ such that

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

Submodular [Nemhauser et al., MP'78]:

$\forall X \subseteq Y \subseteq V, v \notin Y\colon\ f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$;

or $\forall X \subseteq Y \subseteq V\colon\ f(Y) - f(X) \leq \sum_{v \in Y \setminus X} f(X \cup \{v\}) - f(X)$.

Submodular ratio [Das & Kempe, ICML'11; Zhang & Vorobeychi, AAAI'16]:

$$\alpha_f = \min_{X \subseteq Y, v \notin Y} \frac{f(X \cup \{v\}) - f(X)}{f(Y \cup \{v\}) - f(Y)}$$

$\alpha_f = 1$

$\gamma_{U,k}(f) = 1$

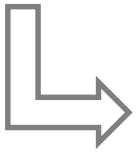
$$\gamma_{U,k}(f) = \min_{X \subseteq U,\, Y:|Y| \leq k, X \cap Y = \emptyset} \frac{\sum_{v \in Y} f(X \cup \{v\}) - f(X)}{f(X \cup Y) - f(X)}$$

Subset selection

Subset selection: given all items $V = \{v_1, \ldots, v_n\}$, an objective function $f\colon 2^V \to \mathrm{R}$ and a budget B, it is to find a subset $X \subseteq V$ such that

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

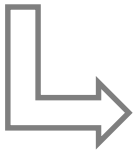
Monotone + Submodular



The optimal approximation guarantee [Nemhauser & Wolsey, MOR'78] :

$1 - 1/e \approx 0.632$ by the greedy algorithm

Monotone



The approximation guarantee [Das & Kempe, ICML'11] :

$1 - 1/e^{\gamma}$ by the greedy algorithm

Variants of subset selection

- Monotone set function maximization with size constraints

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

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

[Das & Kempe, ICML'11]

- Monotone set function maximization with general constraints

$$|X| \leq B \;\rightarrow\; c(X) \leq B$$

$(\alpha/2)\,(1 - 1/e^{\alpha})$

[Zhang & Vorobeychik, AAAI'16]

- Monotone multiset function maximization with size constraints

X: a subset → a multiset

$(\alpha/2)\,(1 - 1/e^{\alpha})$ [Alon et al., WWW'12]

$1 - 1/e^{\beta}$ [Soma et al., ICML'14]

- Monotone k-submodular function maximization with size constraints

X: a subset → k subsets

$1/2$

[Ohsaka & Yoshida, NIPS'15]

- Monotone sequence function maximization with size constraints

X: a subset → a sequence

$1 - e^{-1/(2\Delta)}$

[Tschiatschek et al, AAAI'17]

- Ratio optimization of monotone functions

$$min_{X \subseteq V} \quad f(X)/g(X)$$

$$\frac{|X^*|}{(1 + (|X^*| - 1)(1 - \kappa))\gamma}$$

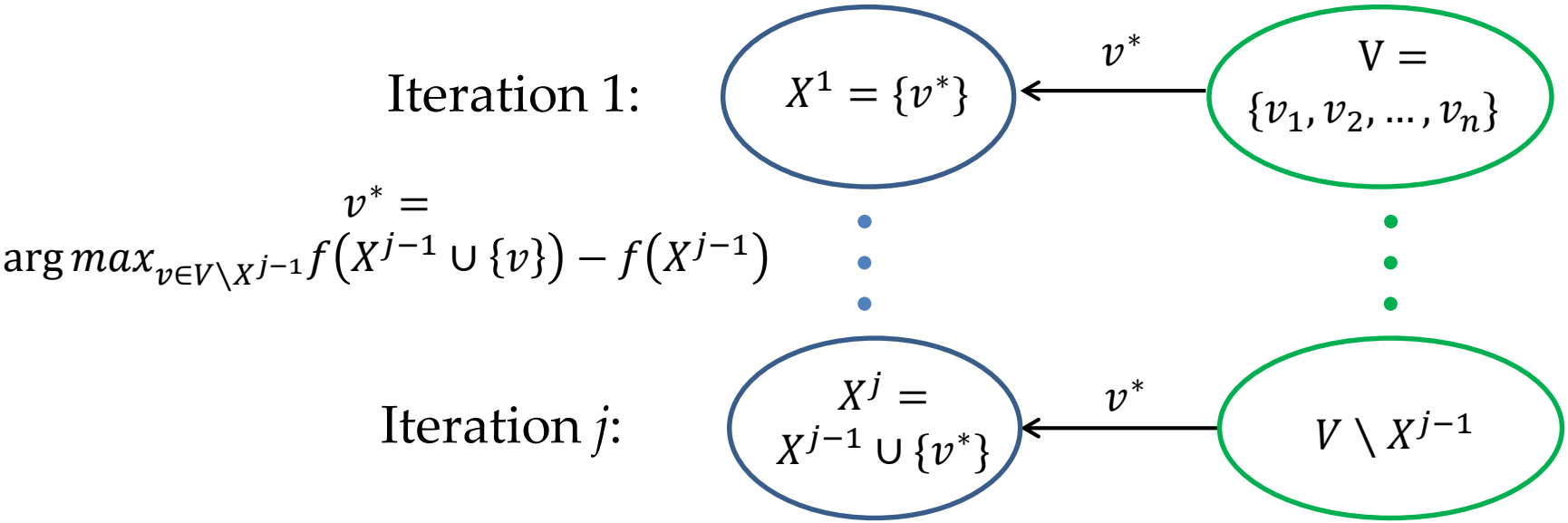
[Bai et al., ICML'16]

Previous approaches

- Greedy algorithms

  Process: iteratively select one item that makes some criterion currently optimized

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

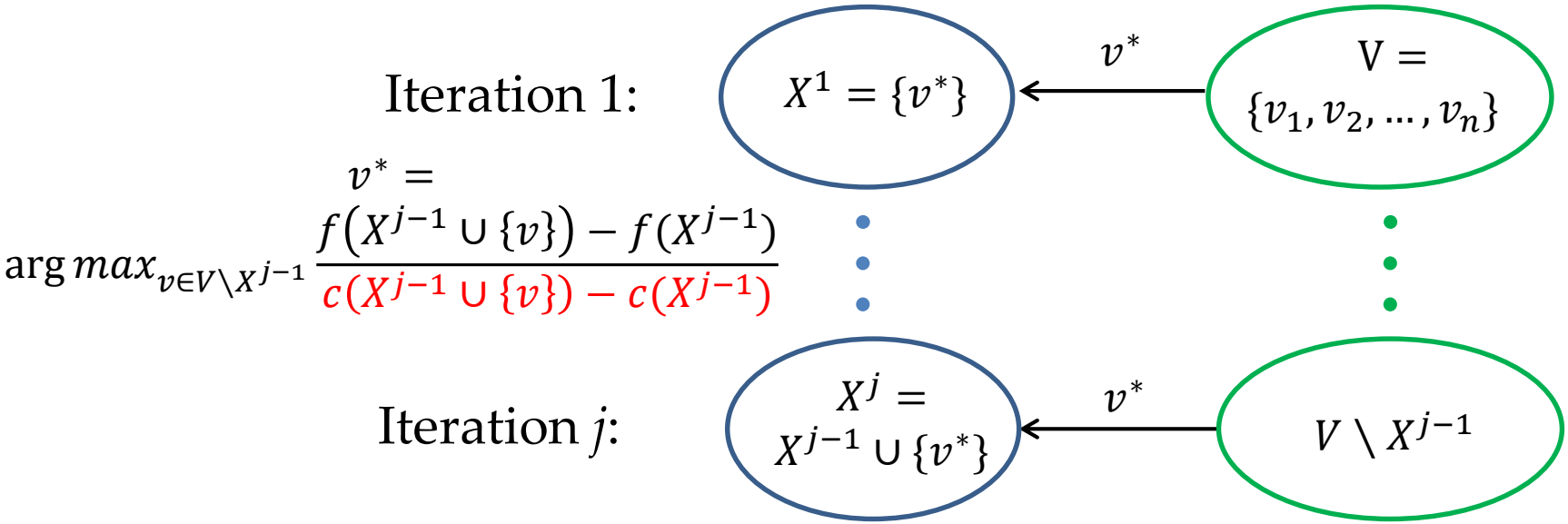


Previous approaches

- Greedy algorithms

  Process: iteratively select one item that makes some criterion currently optimized

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

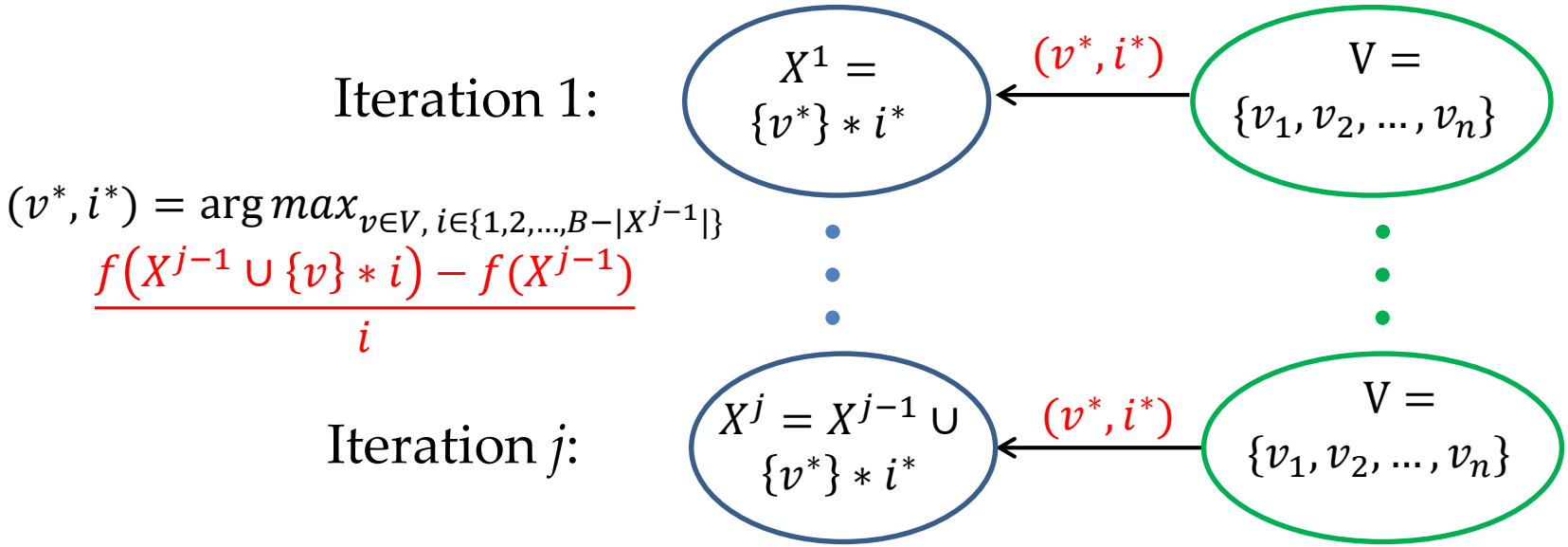


Previous approaches

- Greedy algorithms

Process: iteratively select one item that makes some criterion currently optimized

$$max_{X:a\ multiset\ of\ V} \quad f(X) \quad s.t. \quad |X| \leq B$$

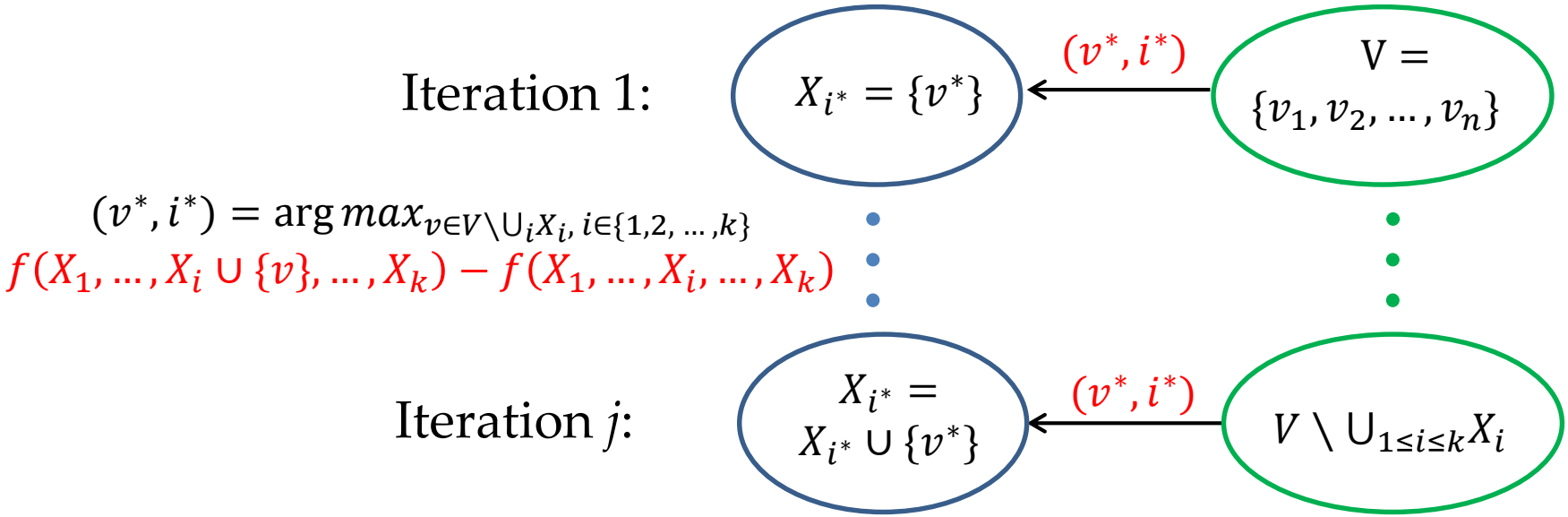


Previous approaches

- Greedy algorithms

Process: iteratively select one item that makes some criterion currently optimized

$$max_{X_1,X_2,\dots,X_k \subseteq V} \quad f(X_1, X_2, \dots, X_k) \;\; s.t. \quad |\cup_{1 \le i \le k} X_i| \le B$$

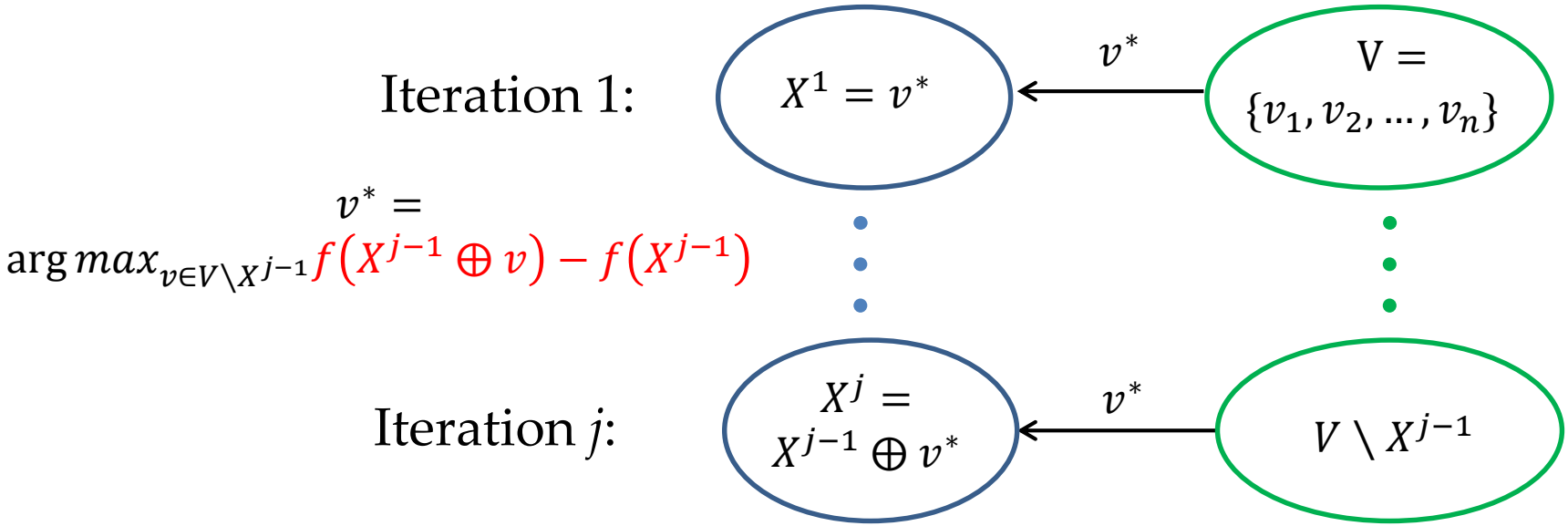


Previous approaches

- Greedy algorithms

  Process: iteratively select one item that makes some criterion currently optimized

$$max_{X:\ a\ sequence\ of\ V}\ f(X)\quad s.t.\quad |X| \le B$$

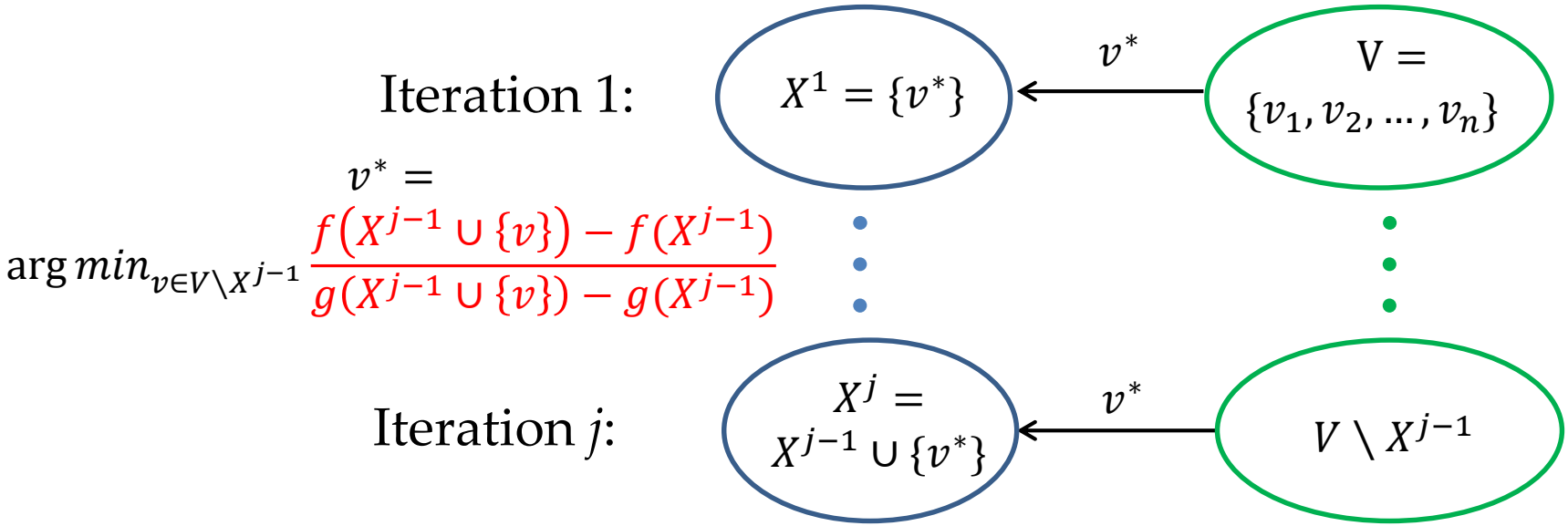


Previous approaches

- Greedy algorithms

  Process: iteratively select one item that makes some criterion currently optimized

$$min_{X \subseteq V} \quad f(X)/g(X)$$

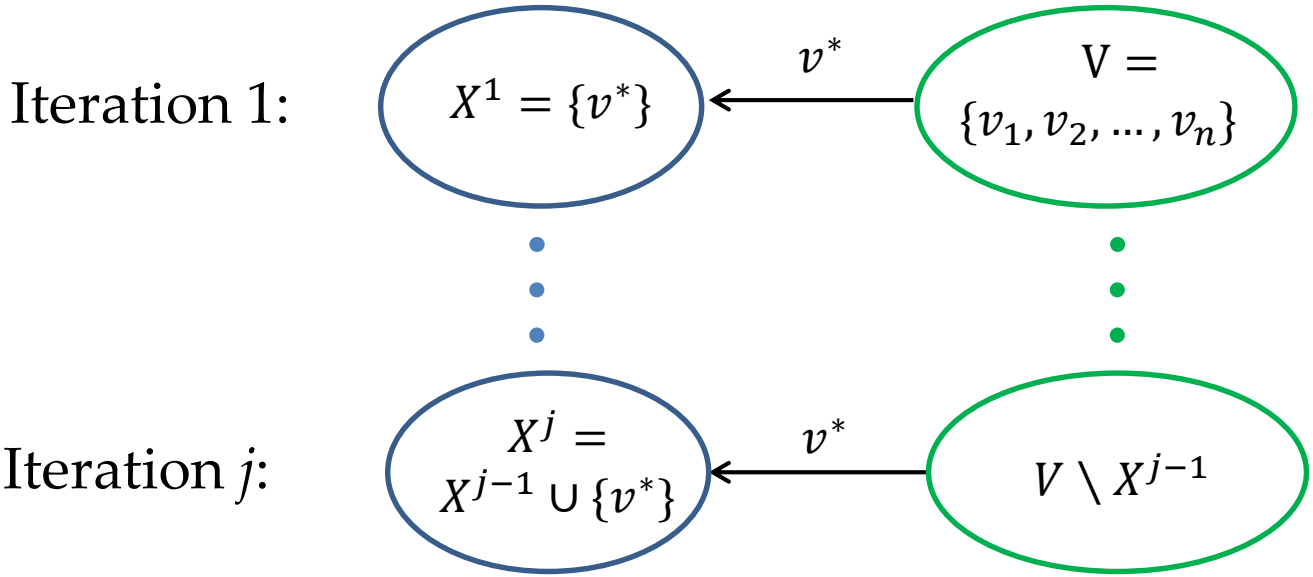


Previous approaches

- Greedy algorithms

  Process: iteratively select one item that makes some criterion currently optimized

  Weakness: get stuck in local optima due to the greedy behavior



Previous approaches (con't)

- Relaxation methods

Process: relax the original problem, then find the optimal solutions to the relaxed problem

Weakness: the optimal solution of the relaxed problem may be distant to the true optimum

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

$$\Updownarrow$$

$$max_{w \in R^n} \quad g(w) \quad s.t. \quad |w|_0 \leq B \qquad \text{non-convex}$$

$$max_{w \in R^n} \quad g(w) \quad s.t. \quad |w|_1 \leq B \qquad \text{convex}$$

Motivation

Subset selection: $max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \ \ |x| \leq B$

a subset $X \subseteq V$

Two conflicting objectives:

1. Optimize the objective f $\quad max_{x \in \{0,1\}^n} \ f(x)$
2. Keep the size small $\quad min_{x \in \{0,1\}^n} \ max\{|x| - B, 0\}$

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

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

Motivation

Subset selection: $max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \ \ |x| \leq B$

a subset $X \subseteq V$

Two conflicting objectives:

1. Optimize the objective f $\quad max_{x \in \{0,1\}^n} \ f(x)$
2. Keep the size small $\quad min_{x \in \{0,1\}^n} \ max\{|x| - B, 0\}$

Why not directly optimize the bi-objective formulation?

$$min_{x \in \{0,1\}^n} \ (-f(x), |x|)$$

Outline

- Introduction
- **Pareto optimization for subset selection**
- Pareto optimization for large-scale subset selection
- Pareto optimization for noisy subset selection
- Conclusion

Pareto optimization

The basic idea:

$max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \quad |x| \leq B$

$max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \quad c(x) \leq B$

$max_{x \in \{0,1,\ldots,k\}^n} \ f(x) \quad s.t. \quad |x| \leq B$

$min_{x \in \{0,1\}^n} \ f(x)/g(x)$

⇨

Bi-objective optimization

$min_x \ (f_1(x), f_2(x))$

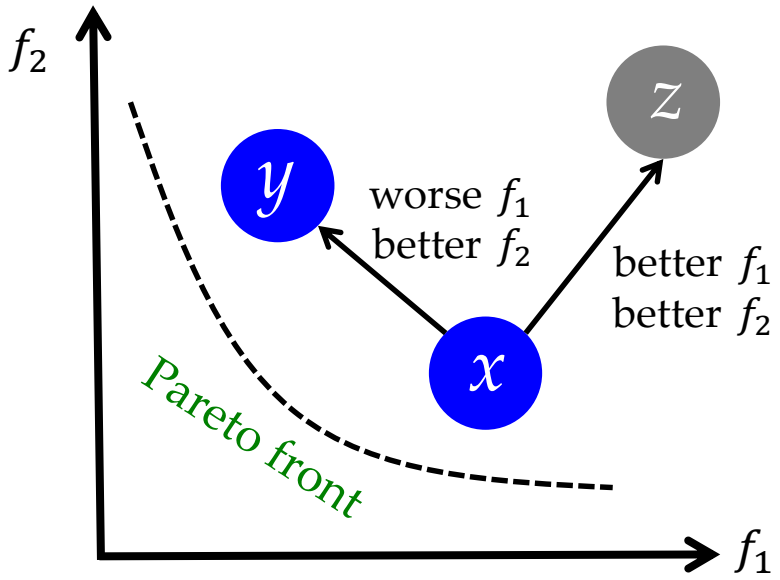
---

x dominates z :

$$f_1(x) < f_1(z) \wedge f_2(x) < f_2(z)$$

x is incomparable with y :

$$f_1(x) > f_1(y) \wedge f_2(x) < f_2(y)$$



Pareto optimization

The basic idea:

$max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \quad |x| \leq B$

$max_{x \in \{0,1\}^n} \ f(x) \quad s.t. \quad c(x) \leq B$

$max_{x \in \{0,1,\dots,k\}^n} \ f(x) \quad s.t. \quad |x| \leq B$

$min_{x \in \{0,1\}^n} \ f(x)/g(x)$

How to transform?

⇨

Bi-objective optimization

$min_x \ (f_1(x), f_2(x))$

⇩

Multi-objective evolutionary algorithms

⇦

Output: select the best solution w.r.t. the original problem

Monotone submodular function maximization with size constraints

[Friedrich and Neumann, ECJ'15]

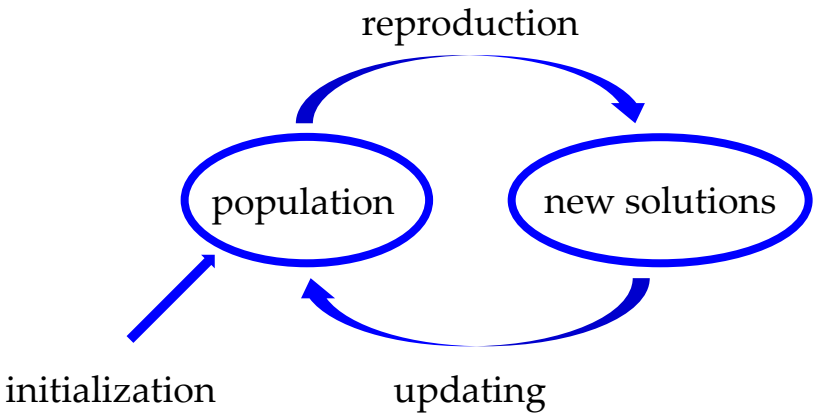
Exclude solutions with size larger than B

$$max_{x\in\{0,1\}^n}\ f(x) \quad s.t. \quad |x| \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1\}^n}\ (-f(x), |x|) \qquad \text{bi-objective}$$

A simple multi-objective evolutionary algorithm GSEMO [Laumanns et al., TEvC'04]



Initialization: put a random solution from $\{0,1\}^n$ into the population P

Reproduction: pick a solution x randomly from P, and flip each bit of $x \in \{0,1\}^n$ with prob. $1/n$ to generate a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best solution with size at most B

It can achieve the optimal approximation guarantee of $1 - 1/e$ in $O(n^2(B + \log n))$ expected running time

Monotone set function maximization with size constraints

The POSS approach [Qian, Yu and Zhou, NIPS'15]

Exclude solutions with size at least $2B$

$$max_{x\in\{0,1\}^n} \ f(x) \quad s.t. \quad |x| \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1\}^n} \ (-f(x), |x|) \qquad \text{bi-objective}$$

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:

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

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

Monotone set function maximization with size constraints

The POSS approach [Qian, Yu and Zhou, NIPS'15]

Exclude solutions with size at least $2B$

$$max_{x\in\{0,1\}^n}\ f(x) \quad s.t. \quad |x| \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1\}^n}\ (-f(x), |x|) \qquad \text{bi-objective}$$

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

- Each solution in P is picked with probability $1/|P|$
- Bit-wise mutation:

$$\Pr(\text{flip } i \text{ specific bits}) = \left(\frac{1}{n}\right)^i \left(1-\frac{1}{n}\right)^{n-i}$$

$$\Pr(\text{flip } i \text{ bits}) = \binom{n}{i}\left(\frac{1}{n}\right)^i \left(1-\frac{1}{n}\right)^{n-i}$$

$$\Pr(x \to x') = \left(\frac{1}{n}\right)^{H(x,x')} \left(1-\frac{1}{n}\right)^{n-H(x,x')}$$

Monotone set function maximization with size constraints

The POSS approach [Qian, Yu and Zhou, NIPS'15]

Exclude solutions with size at least $2B$

$$max_{x\in\{0,1\}^n}\ f(x) \quad s.t. \quad |x| \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1\}^n}\ (-f(x), |x|) \qquad \text{bi-objective}$$

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

- Each solution in P is picked with probability $1/|P|$
- Bit-wise mutation:

$$\Pr(\text{flip } i \text{ specific bits}) = \left(\frac{1}{n}\right)^i \left(1-\frac{1}{n}\right)^{n-i}$$

$$\Pr(\text{flip } i \text{ bits}) = \binom{n}{i}\left(\frac{1}{n}\right)^i \left(1-\frac{1}{n}\right)^{n-i}$$

$$\Pr(x \to x') = \left(\frac{1}{n}\right)^{H(x,x')} \left(1-\frac{1}{n}\right)^{n-H(x,x')}$$

- The population P always contains non-dominated solutions produced so-far

Theoretical analysis

POSS can achieve the same general approximation guarantee as the greedy algorithm

Theorem 1. For monotone set function maximization with size constraints, POSS using $\mathrm{E}[T] \leq 2eB^2n$ finds a solution x with $|x| \leq B$ and $f(x) \geq (1 - e^{-\gamma}) \cdot OPT$.

the expected number of iterations

the best known polynomial-time approximation ratio, previously obtained by the greedy algorithm [Das & Kempe, ICML'11]

Proof

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

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

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

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

Main idea:

$\{0,1\}^n$

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

$i = 0$ → $i = B$

initial solution $00 \ldots 0$

$|00 \ldots 0| = 0$

$f(00 \ldots 0) = 0$

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

the desired approximation bound

Proof

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

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

$\{0,1\}^n$

Main idea:

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

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

Proof

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

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

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

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

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

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

Proof

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

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

$\{0,1\}^n$

Main idea:

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

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

$i \longrightarrow i + 1$ 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

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

$\{0,1\}^n$

Main idea:

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

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

- Exclude solutions with size at least $2B$
- The solutions in P are always incomparable

For each size in $\{0,1,\ldots,2B-1\}$, 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

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

Main idea:

- consider a solution $x \in \{0,1\}^n$ with $|x| \leq i$ and $f(x) \geq \left(1 - \left(1 - \frac{\gamma}{B}\right)^i\right) \cdot OPT$
- in each iteration of POSS:

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

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

$i = 0 \longrightarrow B$ the expected number of iterations: $B \cdot 2eBn$

Theoretical analysis

POSS can achieve the same general approximation guarantee as the greedy algorithm

Theorem 1. For monotone set function maximization with size constraints, POSS using $\mathrm{E}[T] \leq 2eB^2n$ finds a solution x with $|x| \leq B$ and $f(x) \geq (1 - e^{-\gamma}) \cdot OPT$.

the best known polynomial-time approximation ratio, previously obtained by the greedy algorithm [Das & Kempe, ICML'11]

POSS can do better than the greedy algorithm in cases

[Das & Kempe, STOC'08]

Theorem 2. For the Exponential Decay subclass of sparse regression, POSS using $\mathrm{E}[T] = O(B^2(n - B)n \log n)$ finds an optimal solution, while the greedy algorithm cannot.

Sparse regression

Sparse regression [Tropp, TIT'04]: find a sparse approximation solution to the linear regression problem

Formally stated: given all observation variables $V = \{v_1, \ldots, v_n\}$, a predictor variable z and a budget B, it is to find a subset $X \subseteq V$ such that

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

	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

	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 - R^2 values

the size constraint: $B = 8$ **the number of iterations of POSS: $2eB^2n$**

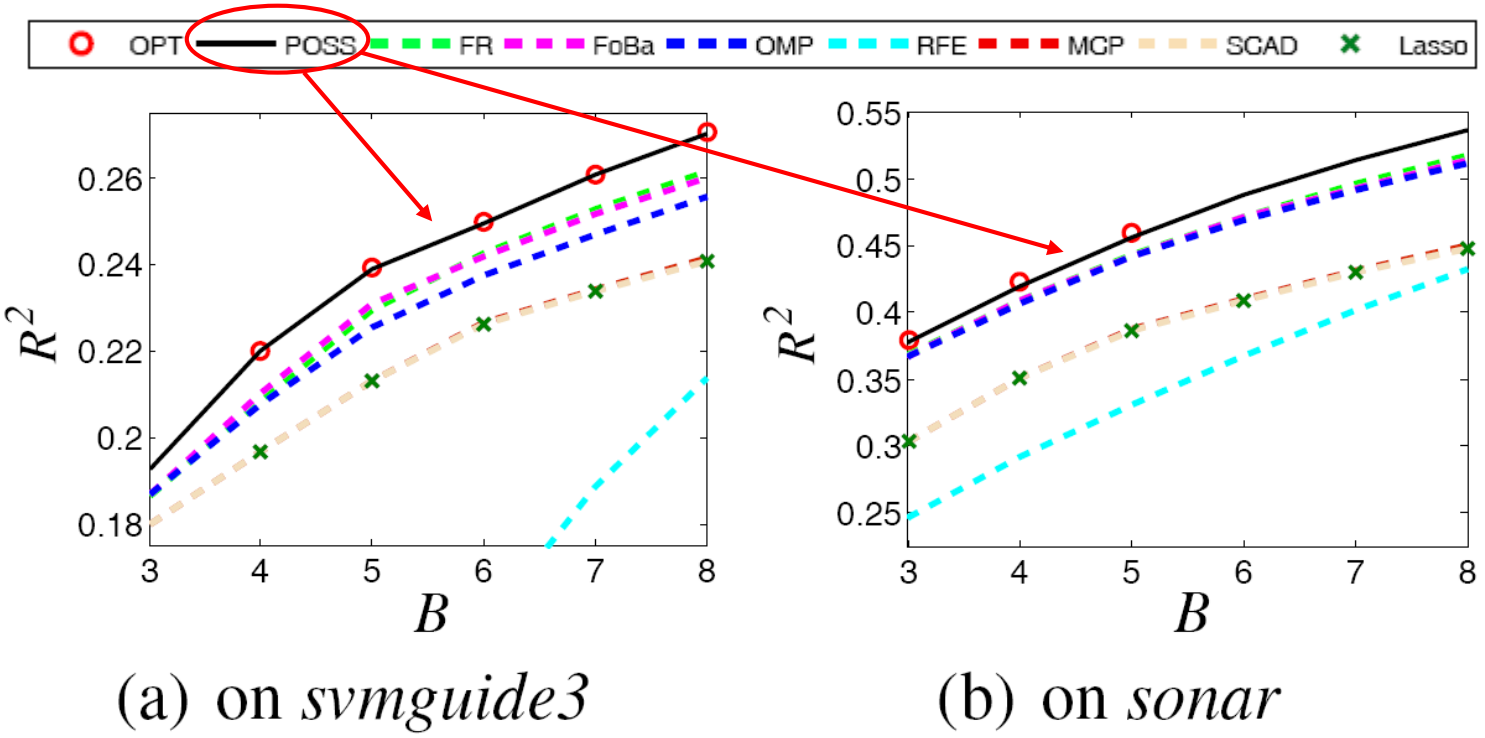
exhaustive search | greedy algorithms | relaxation methods

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

POSS is significantly better than all the compared methods on all data sets

Experimental results - R^2 values

different size constraints: $B = 3 \rightarrow 8$

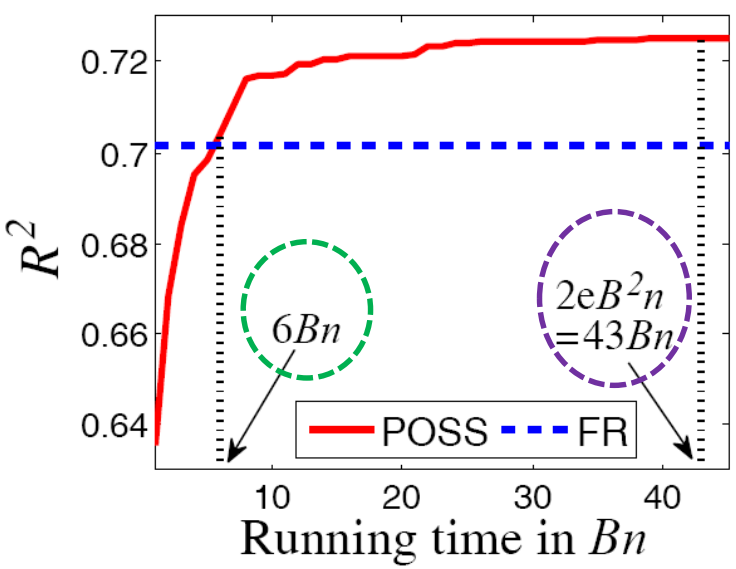


(a) on *svmguide3* (b) on *sonar*

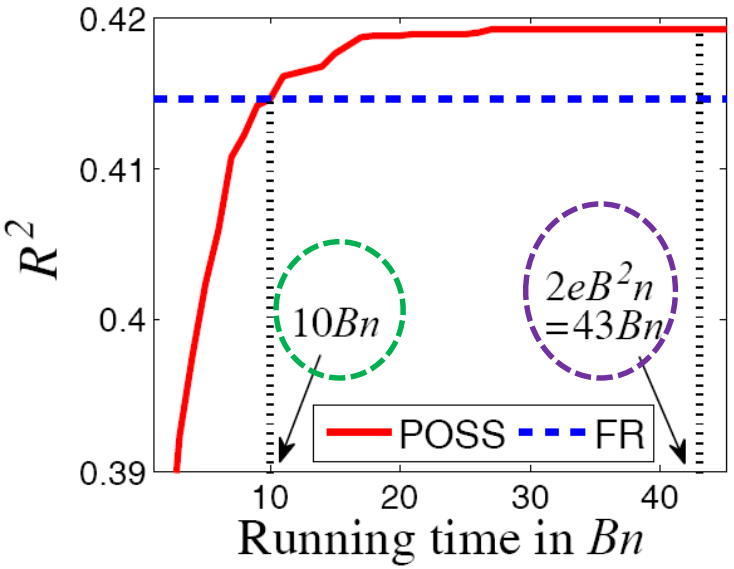
POSS tightly follows OPT, and has a clear advantage over the rest methods

Experimental results – running time

OPT: n^B/B^B greedy methods (FR): Bn POSS: $2eB^2n$



(a) on *gisette*



(b) on *farm-ads*

theoretical running time

POSS can be much more efficient in practice than in theoretical analysis

Pareto optimization vs. Greedy algorithms

Greedy algorithms:

- Produce a new solution by adding a single item (single-bit forward search: $0 \rightarrow 1$)
- Maintain only one solution

Pareto optimization:

- Produce a new solution by flipping each bit of a solution with prob. $1/n$ (single-bit forward search, backward search, multi-bit search)
- Maintain several non-dominated solutions due to bi-objective optimization

Pareto optimization may have a better ability of avoiding local optima!

Monotone set function maximization with general constraints

The POMC approach [Qian, Shi, Yu and Tang, IJCAI'17]

$$max_{x\in\{0,1\}^n}\ f(x) \quad s.t. \quad c(x) \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1\}^n}\ (-f(x), c(x)) \qquad \text{bi-objective}$$

Algorithm 2 POMC Algorithm

Input: a monotone objective function f, a monotone approximate cost function $\hat{c}$, and a budget B
Parameter: the number T of iterations
Output: a solution $\boldsymbol{x} \in \{0,1\}^n$ with $\hat{c}(\boldsymbol{x}) \le B$
Process:

1: Let $\boldsymbol{x} = \{0\}^n$ and $P = \{\boldsymbol{x}\}$.
2: Let $t = 0$.
3: **while** $t < T$ **do**
4: Select $\boldsymbol{x}$ from P uniformly at random.
5: Generate $\boldsymbol{x}'$ by flipping each bit of $\boldsymbol{x}$ with prob. $1/n$.
6: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \succ \boldsymbol{x}'$ **then**
7: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \succeq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$.
8: **end if**
9: $t = t + 1$.
10: **end while**
11: **return** $\arg\max_{\boldsymbol{x}\in P:\hat{c}(\boldsymbol{x})\le B} f(\boldsymbol{x})$

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

Monotone set function maximization with general constraints

The POMC approach [Qian, Shi, Yu and Tang, IJCAI'17]

$max_{x\in\{0,1\}^n}\ f(x) \quad s.t. \quad c(x) \le B$ original

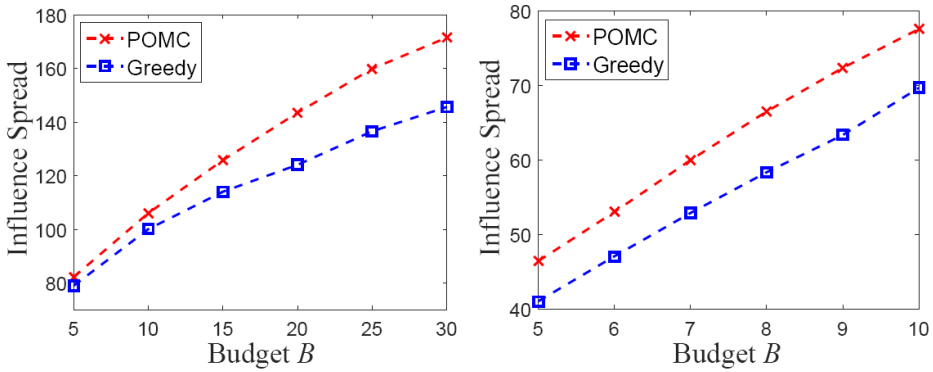
Transformation: ⇩

$min_{x\in\{0,1\}^n}\ (-f(x), c(x))$ bi-objective

Theory: POMC can achieve the same approximation guarantee $(\alpha/2)(1-e^{-\alpha})$ as the greedy algorithm [Zhang & Vorobeychik, AAAI'16]

Application:

influence maximization



(a) (Digg, cardinality) (b) (Synthetic, routing)

Monotone multiset function maximization with size constraints

The POMS approach [Qian, Zhang, Tang and Yao, AAAI'18]

Transformation:

$$max_{x \in Z_+^n} \ f(x) \quad s.t. \quad |x| \leq B \qquad \text{original}$$

$$\Downarrow$$

$$min_{x \in Z_+^n} \ (-f(x), |x|) \qquad \text{bi-objective}$$

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

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a multiset $X \subseteq V$ $\{v_2, v_2, v_3, v_5, v_5\}$ $\Leftrightarrow$ an integer vector $x \in Z_+^5$ 02102

Monotone multiset function maximization with size constraints

The POMS approach [Qian, Zhang, Tang and Yao, AAAI'18]

$$max_{x \in \mathbb{Z}_+^n} \ f(x) \quad s.t. \quad |x| \leq B \qquad \text{original}$$

Transformation: ⇩

$$min_{x \in \mathbb{Z}_+^n} \ (-f(x), |x|) \qquad \text{bi-objective}$$

Algorithm 1 POMS Algorithm

Input: a monotone function $f : \mathbb{Z}_+^V \to \mathbb{R}_+$, a vector $\boldsymbol{c} \in \mathbb{Z}_+^V$ and a budget $k \in \mathbb{Z}_+$
Parameter: the number T of iterations
Output: a multiset $\boldsymbol{x} \in \mathbb{Z}_+^V$ with $\boldsymbol{x} \leq \boldsymbol{c}$ and $|\boldsymbol{x}| \leq k$
Process:

1: Let $\boldsymbol{x} = \mathbf{0}$ and $P = \{\boldsymbol{x}\}$.
2: Let $t = 0$.
3: **while** $t < T$ **do**
4: Select $\boldsymbol{x}$ from P uniformly at random.
5: $\boldsymbol{x}' = RandomPerturbation(\boldsymbol{x})$.
6: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \succ \boldsymbol{x}'$ **then**
7: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \succeq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$.
8: **end if**
9: $t = t + 1$.
10: **end while**
11: **return** $\arg\max_{\boldsymbol{x} \in P : |\boldsymbol{x}| \leq k} f(\boldsymbol{x})$

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

Monotone multiset function maximization with size constraints

The POMS approach [Qian, Zhang, Tang and Yao, AAAI'18]

Transformation:

$max_{x \in \mathbb{Z}_+^n} \ f(x) \quad s.t. \quad |x| \leq B$ original

$\Downarrow$

$min_{x \in \mathbb{Z}_+^n} \ (-f(x), |x|)$ bi-objective

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

flip on one position: 0 → 1 or 1 → 0

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a multiset $X \subseteq V$ $\{v_2, v_2, v_3, v_5, v_5\}$ $\Leftrightarrow$ an integer vector $x \in Z_+^5$ 02102

flip on one position: x_i → a value selected from $\{0,1, \ldots, c_i\} \backslash \{x_i\}$ randomly

Monotone multiset function maximization with size constraints

The POMS approach [Qian, Zhang, Tang and Yao, AAAI'18]

$$max_{x \in \mathbb{Z}_+^n} \ f(x) \quad s.t. \quad |x| \leq B \qquad \text{original}$$

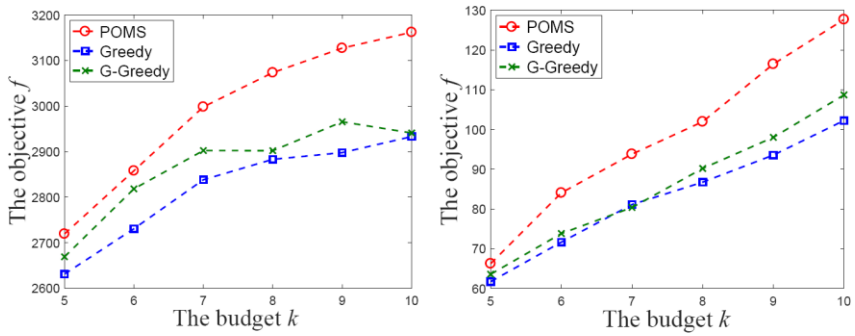
Transformation: ⇩

$$min_{x \in \mathbb{Z}_+^n} \ (-f(x), |x|) \qquad \text{bi-objective}$$

Theory: POMS can achieve the approximation guarantee of $\max\{1 - 1/e^{\beta}, (\alpha/2)(1 - 1/e^{\alpha})\}$, which is the better of two greedy algorithms [Alon et al., WWW'12; Soma et al., ICML'14]

Application:

generalized influence maximization



(a) *ego-Facebook* (b) *Weibo*

Monotone k-submodular function maximization with size constraints

The MOMS approach [Qian, Shi, Tang and Zhou, TEvC in press]

Transformation:

$max_{x\in\{0,1,\dots,k\}^n}\ f(x)\ \ s.t.\ |x| \le B$ original

⇩

$min_{x\in\{0,1,\dots,k\}^n}\ (-f(x), |x|)$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

$\{v_2, v_3, v_5\}$ ⇔ 01101

---

k subsets $X_1, X_2, \dots, X_k \subseteq V$ ⇔ an integer vector $x \in \{0,1,\dots,k\}^5$

$\{v_3\}, \{v_2, v_4, v_5\}$ ⇔ 02122

The MOMS approach [Qian, Shi, Tang and Zhou, TEvC in press]

$$max_{x\in\{0,1,\ldots,k\}^n}\ f(x)\ \ s.t.\ |x| \le B \qquad \text{original}$$

Transformation: ⇩

$$min_{x\in\{0,1,\ldots,k\}^n}\ (-f(x), |x|) \qquad \text{bi-objective}$$

Algorithm 2 MOMS

Input: a monotone k-submodular function $f : \{0,\ldots,k\}^n \to \mathbb{R}^+$ and a budget $b \in [n]$
Parameter: the number T of iterations
Output: a solution $\boldsymbol{x}$ with $|supp(\boldsymbol{x})| \le b$
Process:

1: Let $\boldsymbol{x} = \mathbf{0}$ and $P = \{\boldsymbol{x}\}$.
2: Let $t = 0$.
3: **while** $t < T$ **do**
4: Select $\boldsymbol{x}$ from P uniformly at random.
5: $\boldsymbol{y} = Mutation(\boldsymbol{x})$.
6: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \prec \boldsymbol{y}$ **then**
7: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{y} \preceq \boldsymbol{z}\}) \cup \{\boldsymbol{y}\}$.
8: $Q = RLS(f, b, \boldsymbol{y})$.
9: **for** each $\boldsymbol{q} \in Q$
10: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \prec \boldsymbol{q}$ **then**
11: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{q} \preceq \boldsymbol{z}\}) \cup \{\boldsymbol{q}\}$.
12: **end if**
13: **end for**
14: **end if**
15: $t = t + 1$.
16: **end while**
17: **return** $\arg\min_{\boldsymbol{x}\in P, |supp(\boldsymbol{x})|\le b} f_1(\boldsymbol{x})$

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

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions; further perform randomized local search on the new solution to improve P

Output: select the best feasible solution

Monotone k-submodular function maximization with size constraints

The MOMS approach [Qian, Shi, Tang and Zhou, TEvC in press]

Transformation:

$max_{x\in\{0,1,\dots,k\}^n}\ f(x)\ \ s.t.\ |x| \le B$ original

⇩

$min_{x\in\{0,1,\dots,k\}^n}\ (-f(x), |x|)$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

$\{v_2, v_3, v_5\}$ ⇔ 01101

flip on one position: 0 → 1 or 1 → 0

---

k subsets $X_1, X_2, \dots, X_k \subseteq V$ ⇔ an integer vector $x \in \{0,1,\dots,k\}^5$

$\{v_3\}, \{v_2, v_4, v_5\}$ ⇔ 02122

flip on one position: x_i → a value selected from $\{0,1,\dots,k\}\backslash\{x_i\}$ randomly

Monotone k-submodular function maximization with size constraints

The MOMS approach [Qian, Shi, Tang and Zhou, TEvC in press]

$$max_{x\in\{0,1,\ldots,k\}^n} \ f(x) \ \ s.t. \ |x| \leq B \qquad \text{original}$$

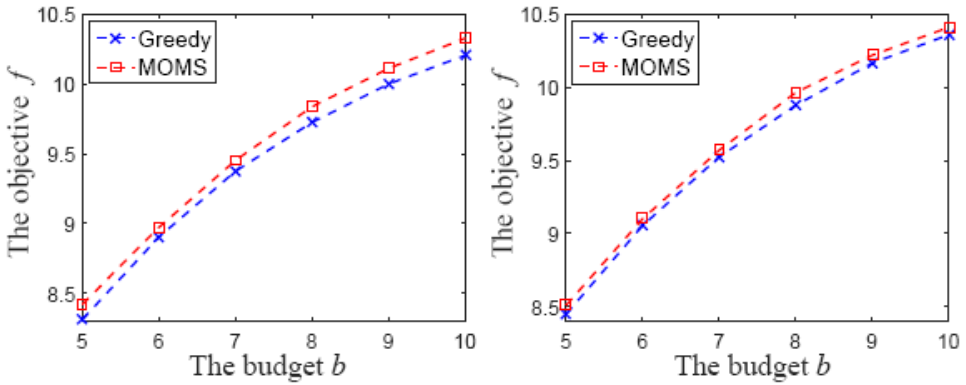
Transformation: ⇩

$$min_{x\in\{0,1,\ldots,k\}^n} \ (-f(x), |x|) \qquad \text{bi-objective}$$

Theory: MOMS can achieve the same approximation guarantee 1/2 as the greedy algorithm [Ohsaka & Yoshida, NIPS'15]

Application:

sensor placement



Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B$$ original

Transformation: ⇩

$$min_{x \in \mathcal{S}} \ (-f(x), |x|)$$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

---

a sequence $x \in \mathcal{S} = \{(x_1, x_2, \ldots, x_l) \mid x_i \in V, l \in \mathrm{Z}^+\}$ $v_2 v_5 v_4$

Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B \qquad \text{original}$$

Transformation: ⇩

$$min_{x \in \mathcal{S}} \ (-f(x), |x|) \qquad \text{bi-objective}$$

Algorithm 1 POSEQSEL Algorithm

Input: all items $V = \{v_1, v_2, \ldots, v_n\}$, a sequence function $f : \mathcal{S} \to \mathbb{R}$ and a budget $k \in \mathbb{Z}^+$
Parameter: the number T of iterations
Output: a sequence $s \in \mathcal{S}$ with $|s| \leq k$
Process:

1: Let $P = \{\emptyset\}$ and $t = 0$.
2: **while** $t < T$ **do**
3: Select a sequence s from P uniformly at random.
4: r = a random number sampled from the Poisson distribution with $\lambda = 1$.
5: $s' = s$.
6: **for** $i = 1$ to r
7: $s' = insert(s')$ or $delete(s')$, each with prob. 1/2.
8: **end for**
9: **if** $\nexists t \in P$ such that $t \succ s'$ **then**
10: $P = (P \setminus \{t \in P \mid s' \succeq t\}) \cup \{s'\}$.
11: **end for**
12: $t = t + 1$.
13: **end while**
14: **return** $\arg\max_{s \in P : |s| \leq k} f(s)$

Initialization: put the special solution $\emptyset$ into the population P

Reproduction: pick a solution x randomly from P, and mutate x to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions

Output: select the best feasible solution

Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B$ original

Transformation: ⇩

$min_{x \in \mathcal{S}} \ (-f(x), |x|)$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

Bit-wise mutation: flip each bit of a solution with prob. $1/n$

Bit-wise mutation: select a number r randomly from the binomial distribution $B(n, 1/n)$, then flip r randomly chosen bits

---

a sequence $x \in \mathcal{S}$ $v_2 v_5 v_4$ → insertion: $v_2 v_5 v_3 v_4$

→ deletion: $v_5 v_4$

Mutation: select a number r randomly from the Poisson distribution with $\lambda = 1$, then perform insertion or deletion uniformly at random for r times

Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B$ original

Transformation: ⇩

$min_{x \in \mathcal{S}} \ (-f(x), |x|)$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

Bit-wise mutation: flip each bit of a solution with prob. $1/n$

Bit-wise mutation: select a number r randomly from the binomial distribution $B(n, 1/n)$, then flip r randomly chosen bits

---

a sequence $x \in \mathcal{S}$ $v_2 v_5 v_4$ → insertion: $v_2 v_5 v_3 v_4$

→ deletion: $v_5 v_4$

Mutation: select a number r randomly from the Poisson distribution with $\lambda = 1$, then perform insertion or deletion uniformly at random for r times

Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B$ original

Transformation: ⇩

$min_{x \in \mathcal{S}} \ (-f(x), |x|)$ bi-objective

$V = \{v_1, v_2, v_3, v_4, v_5\}$ a subset $X \subseteq V$ ⇔ a Boolean vector $x \in \{0,1\}^5$

Bit-wise mutation: flip each bit of a solution with prob. $1/n$

Bit-wise mutation: select a number r randomly from the binomial distribution $B(n, 1/n)$, then flip r randomly chosen bits

---

a sequence $x \in \mathcal{S}$ $v_2 v_5 v_4$ → insertion: $v_2 v_5 v_3 v_4$

→ deletion: $v_5 v_4$

Mutation: select a number r randomly from the Poisson distribution with $\lambda = 1$, then perform insertion or deletion uniformly at random for r times

Monotone sequence function maximization with size constraints

The POSeqSel approach [Qian, Feng and Tang, IJCAI'18]

$max_{x \in \mathcal{S}} \ f(x) \quad s.t. \quad |x| \leq B$ original

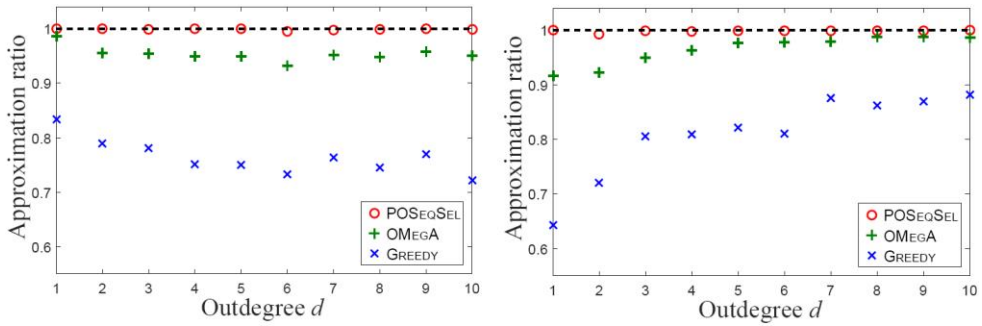
Transformation: ⇩

$min_{x \in \mathcal{S}} \ (-f(x), |x|)$ bi-objective

Theory: POSeqSel can achieve the approximation guarantee $1 - e^{-1/2}$, which is better than that of the greedy algorithm, i.e., $1 - e^{-1/(2\Delta)}$ where $\Delta \geq 1$ [Tschiatschek et al., AAAI'17]

Application:

movie recommendation



Ratio optimization of monotone functions

The PORM approach [Qian, Shi, Yu, Tang and Zhou, IJCAI'17]

Transformation:

$$min_{x\in\{0,1\}^n} \ f(x)/g(x) \qquad \text{original}$$

$$\Downarrow$$

$$min_{x\in\{0,1\}^n} \ (f(x), -g(x)) \qquad \text{bi-objective}$$

Algorithm 2 PORM algorithm

Input: a monotone submodular function $f : \{0,1\}^n \to \mathbb{R}^+$ and a monotone function $g : \{0,1\}^n \to \mathbb{R}^+$
Parameter: the number T of iterations
Output: a solution $\boldsymbol{x} \in \{0,1\}^n$
Process:

1: Select $\boldsymbol{x}$ from $\{0,1\}^n$ uniformly at random.
2: Let $P = \{\boldsymbol{x}\}$ and $t = 0$.
3: **while** $t < T$ **do**
4: Select $\boldsymbol{x}$ from P uniformly at random.
5: Generate $\boldsymbol{x}'$ by flipping each bit of $\boldsymbol{x}$ with prob. $1/n$.
6: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \prec \boldsymbol{x}'$ **then**
7: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \preceq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$.
8: $Q = \{\boldsymbol{z} \in P \mid |\boldsymbol{z}| = |\boldsymbol{x}'|\}$.
9: $\boldsymbol{z}_1 = \arg\min_{\boldsymbol{z}\in Q} f(\boldsymbol{z})$, $\boldsymbol{z}_2 = \arg\max_{\boldsymbol{z}\in Q} g(\boldsymbol{z})$, $\boldsymbol{z}_3 = \arg\min_{\boldsymbol{z}\in Q} f(\boldsymbol{z})/g(\boldsymbol{z})$.
10: $P = (P \setminus Q) \cup \{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3\}$.
11: **end if**
12: $t = t + 1$.
13: **end while**
14: **return** $\arg\min_{\boldsymbol{x}\in P} f(\boldsymbol{x})/g(\boldsymbol{x})$

Initialization: put a random solution from $\{0,1\}^n$ into the population P

Reproduction: pick a solution x randomly from P, and flip each bit of x with prob. $1/n$ to produce a new solution

Updating: if the new solution is not dominated by any solution in P, put it into P and weed out bad solutions; keep at most three solutions for each subset size

Output: select the best feasible solution

Ratio optimization of monotone functions

The PORM approach [Qian, Shi, Yu, Tang and Zhou, IJCAI'17]

Transformation:

$min_{x\in\{0,1\}^n}\ f(x)/g(x)$ original

⇩

$min_{x\in\{0,1\}^n}\ (f(x), -g(x))$ bi-objective

Theory: PORM can achieve the same approximation guarantee $\frac{|X^*|}{(1+(|X^*|-1)(1-\kappa))\gamma}$ as the greedy algorithm [Bai et al., ICML'16]

Application:

F-measure maximization in information retrieval

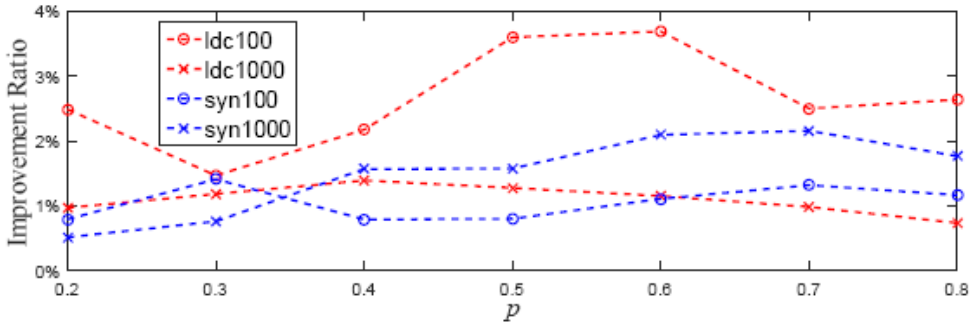


Figure 1: Ratio of improvement of PORM to GreedRatio.

Pareto optimization for subset selection

achieve superior performance on diverse variants of subset selection both theoretically and empirically

The running time (e.g., $2eB^2n$) for achieving a good solution

unsatisfactory when the problem size (e.g., B and n) is large

A sequential algorithm that cannot be readily parallelized

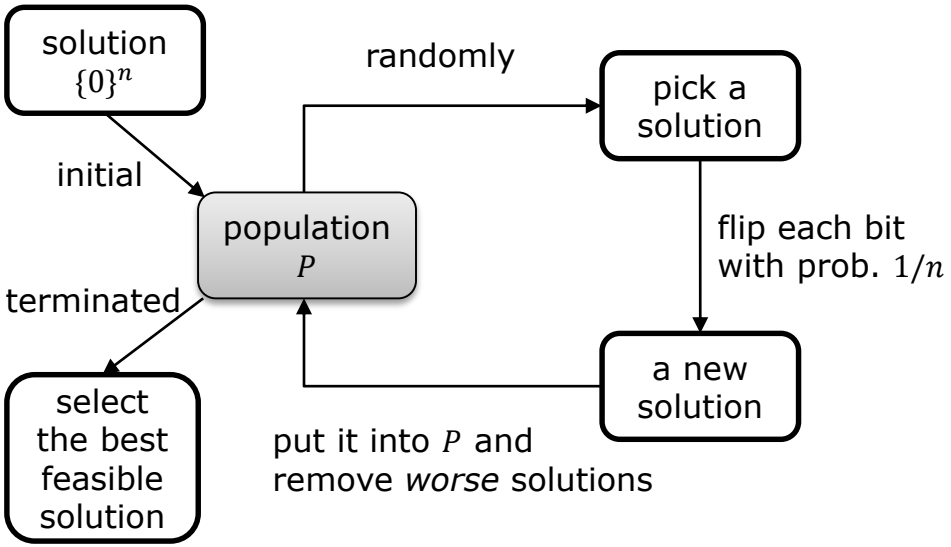
restrict the application to large-scale real-world problems

Can we make the Pareto optimization method parallelizable?

Outline

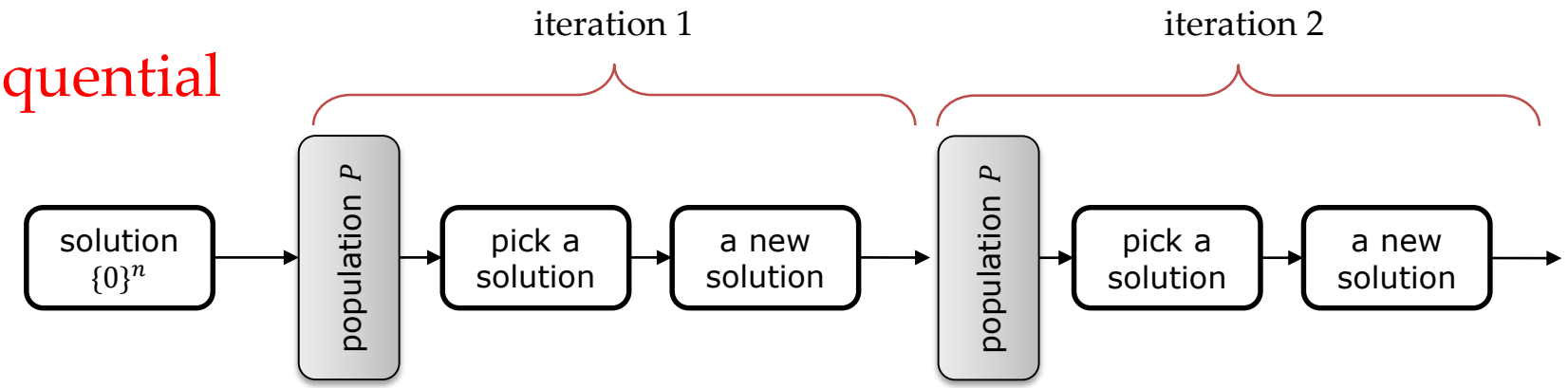
- Introduction
- Pareto optimization for subset selection
- **Pareto optimization for large-scale subset selection**
- Pareto optimization for noisy subset selection
- Conclusion

Pareto optimization for subset selection

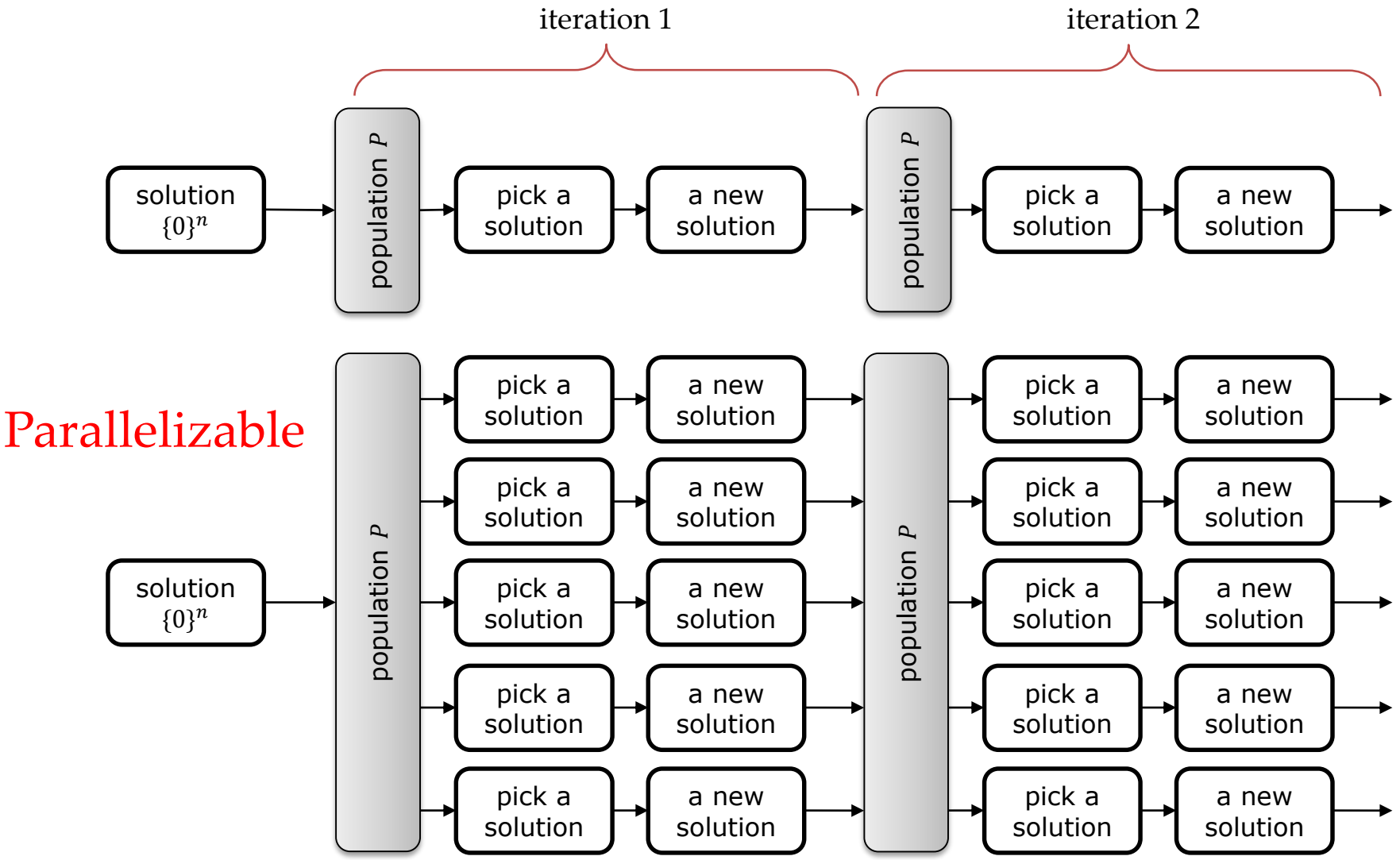


1. put the special solution $\{0\}^n$ into the population P;
2. loop
   - 2.1 pick a solution randomly from P;
   - 2.2 randomly change it to make a new one;
   - 2.3 if the new one is not *strictly worse*
     - 2.3.1 put it into P;
     - 2.3.2 remove *worse* solutions from P;
3. when terminating, select the best feasible solution from P.

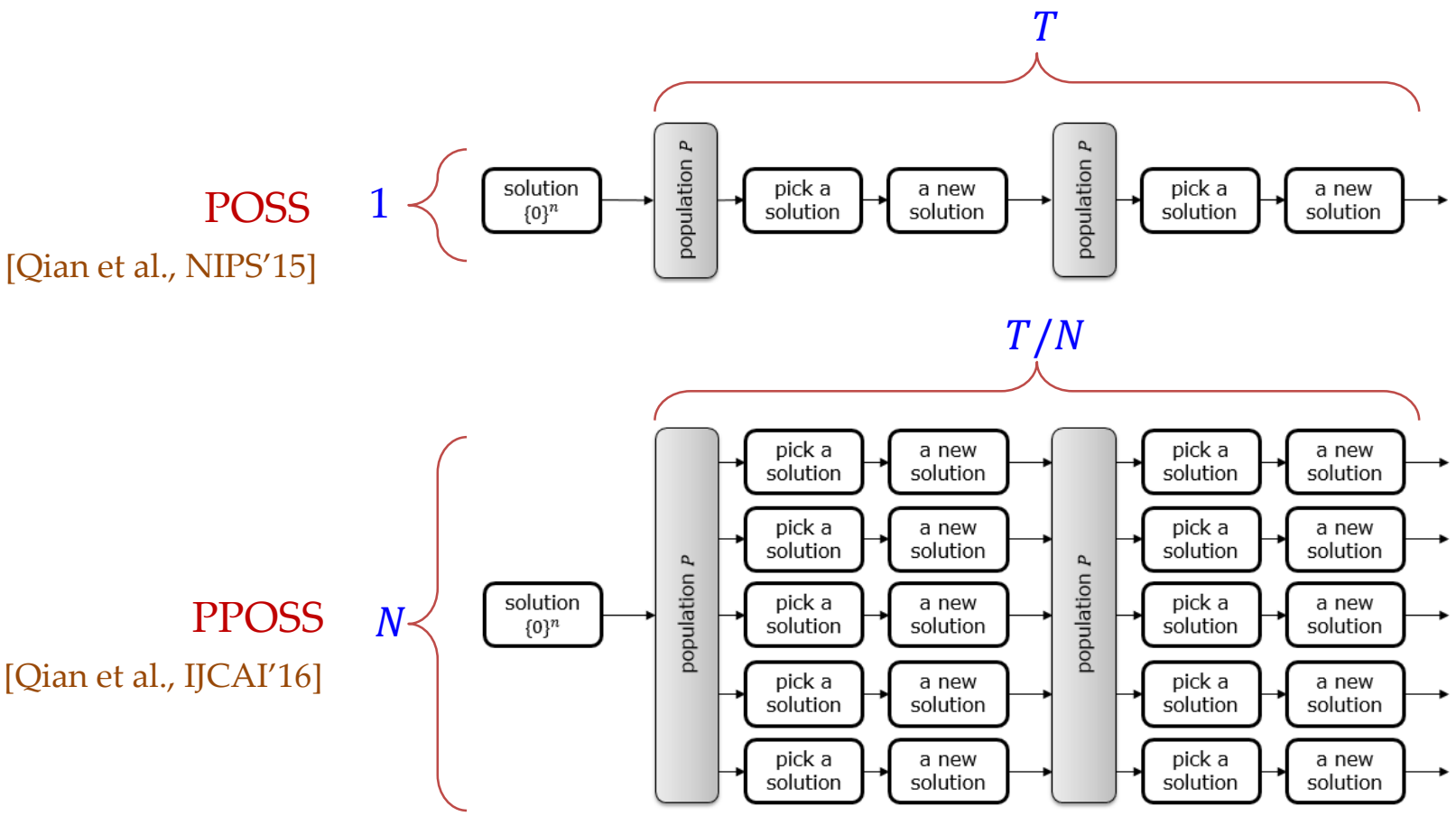
Sequential



Parallel Pareto optimization for subset selection



Parallel Pareto optimization for subset selection

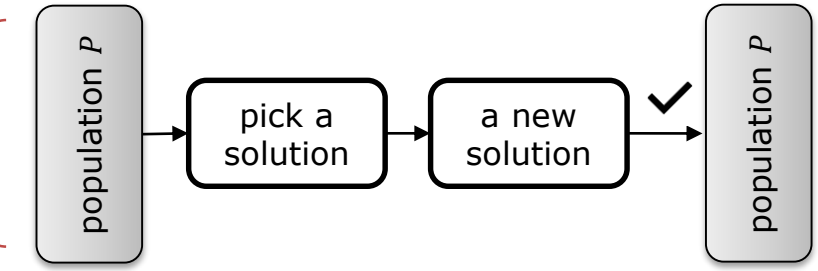


Q: the same solution quality? Yes!

Parallel Pareto optimization for subset selection

POSS

[Qian et al., NIPS'15]

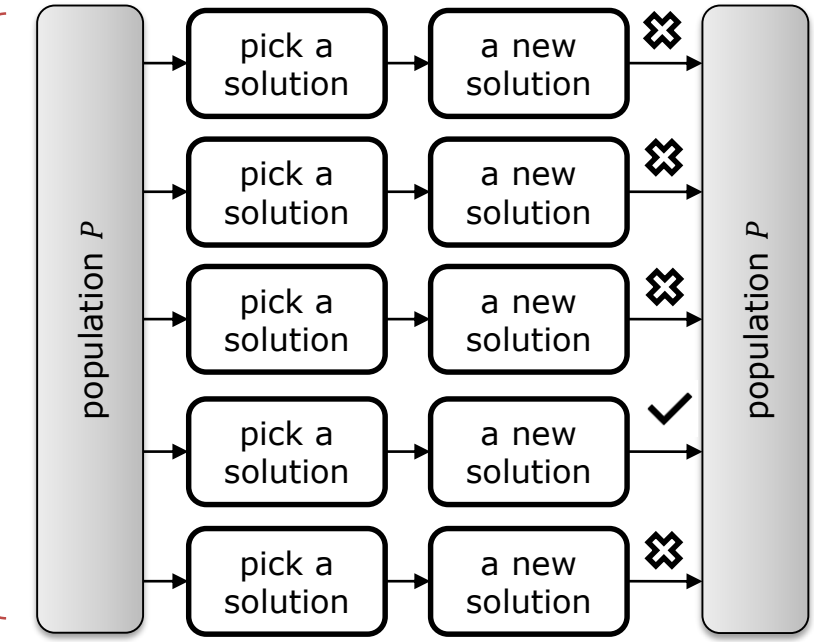


$\frac{1}{en}$

Linear speedup

PPOSS

[Qian et al., IJCAI'16]



$\frac{1}{en}$

$1-\frac{1}{en}$

$\left(1-\frac{1}{en}\right)^N$

$1-\left(1-\frac{1}{en}\right)^N \approx \frac{N}{en}$

Theoretical analysis

Theorem 3. For monotone set function maximization with size constraints, the expected number of iterations until PPOSS finds a solution x with $|x| \leq B$ and $f(x) \geq (1 - e^{-\gamma}) \cdot OPT$ is

(1) if $N = o(n)$, then $\mathrm{E}[T] \leq 2eB^2 n/N$;

(2) if $N = \Omega(n^i)$ for $1 \leq i \leq B$, then $\mathrm{E}[T] = O(B^2/i)$;

(3) if $N = \Omega(n^{\min\{3B-1,n\}})$, then $\mathrm{E}[T] = O(1)$.

the same approximation bound

- When the number N of processors is less than the number n of items, the number T of iterations can be reduced linearly w.r.t. the number of processors

Theoretical analysis

Theorem 3. For monotone function maximization with cardinality constraints, the expected number of iterations until PPOSS finds a solution x with $|x| \leq B$ and $f(x) \geq (1 - e^{-\gamma}) \cdot OPT$ is

(1) if $N = o(n)$, then $\mathrm{E}[T] \leq 2eB^2 n/N$;

(2) if $N = \Omega(n^i)$ for $1 \leq i \leq B$, then $\mathrm{E}[T] = O(B^2/i)$;

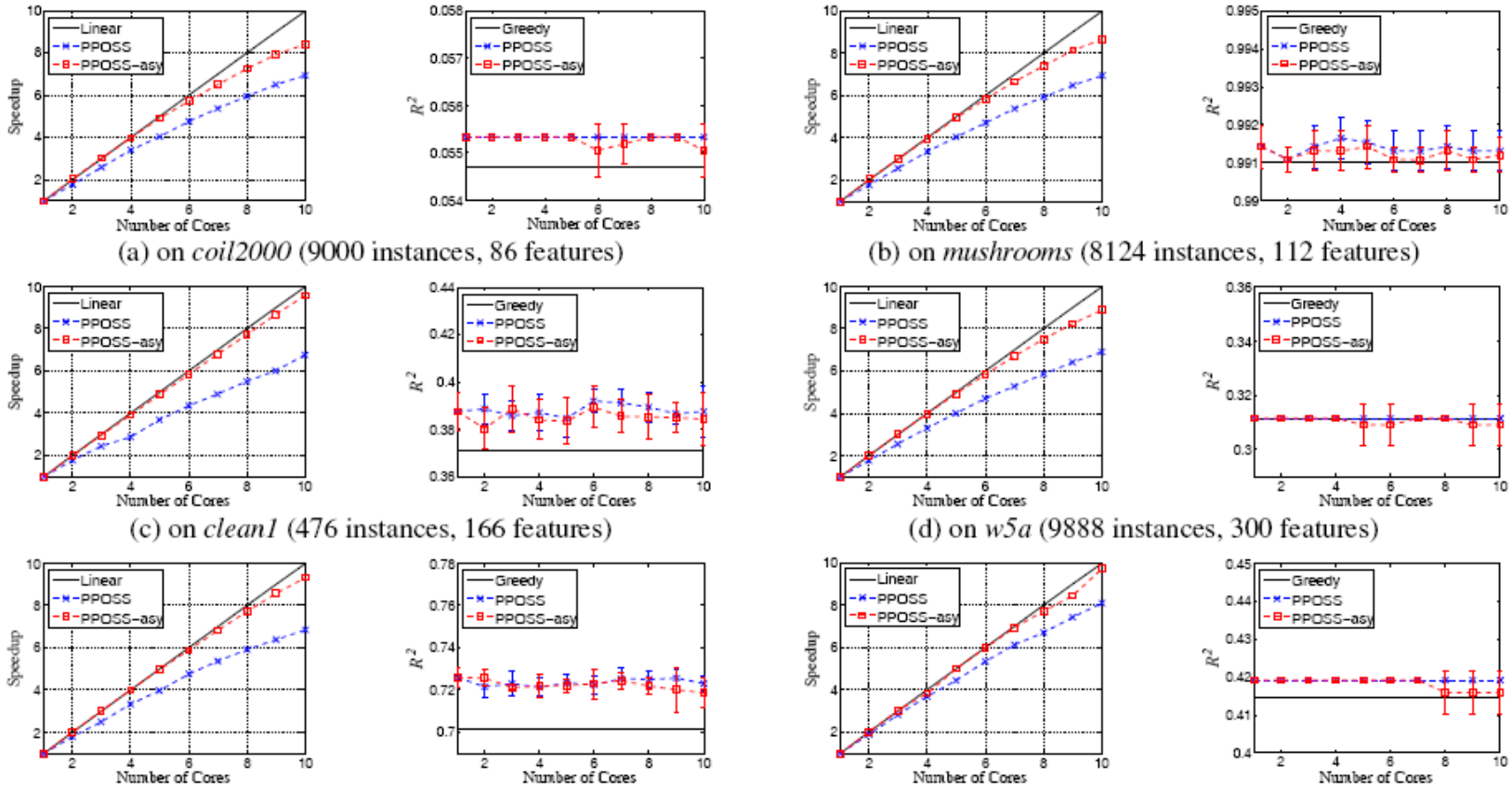
(3) if $N = \Omega(n^{\min\{3B-1,n\}})$, then $\mathrm{E}[T] = O(1)$.

the same approximation bound

- When the number N of processors is less than the number n of items, the number T of iterations can be reduced linearly w.r.t. the number of processors
- With increasing number N of processors, the number T of iterations can be continuously reduced, eventually to a constant

Experiments on sparse regression

Compare the speedup as well as the solution quality measured by R^2 values with different number of cores



(a) on *coil2000* (9000 instances, 86 features)

(b) on *mushrooms* (8124 instances, 112 features)

(c) on *clean1* (476 instances, 166 features)

(d) on *w5a* (9888 instances, 300 features)

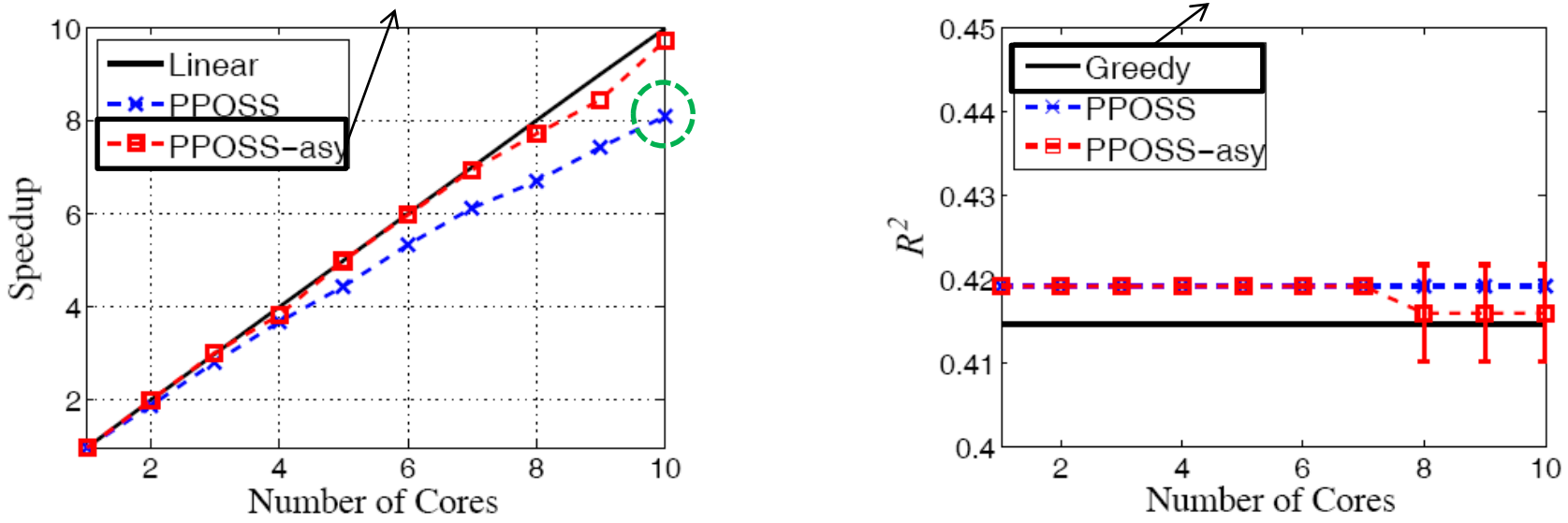
(e) on *gisette* (7000 instances, 5000 features)

(f) on *farm-ads* (4143 instances, 54877 features)

Experiments on sparse regression

the asynchronous version of PPOSS

the best previous algorithm [Das & Kempe, ICML'11]



(f) on *farm-ads* (4143 instances, 54877 features)

PPOSS (blue line): achieve speedup around 8 when the number of cores is 10; the R^2 values are stable, and better than Greedy

PPOSS-asy (red line): achieve better speedup (avoid the synchronous cost)**; the R^2 values are slightly worse** (the noise from asynchronization)

Pareto optimization for subset selection

achieve superior performance on diverse variants of subset selection both theoretically and empirically

Parallel Pareto optimization for subset selection

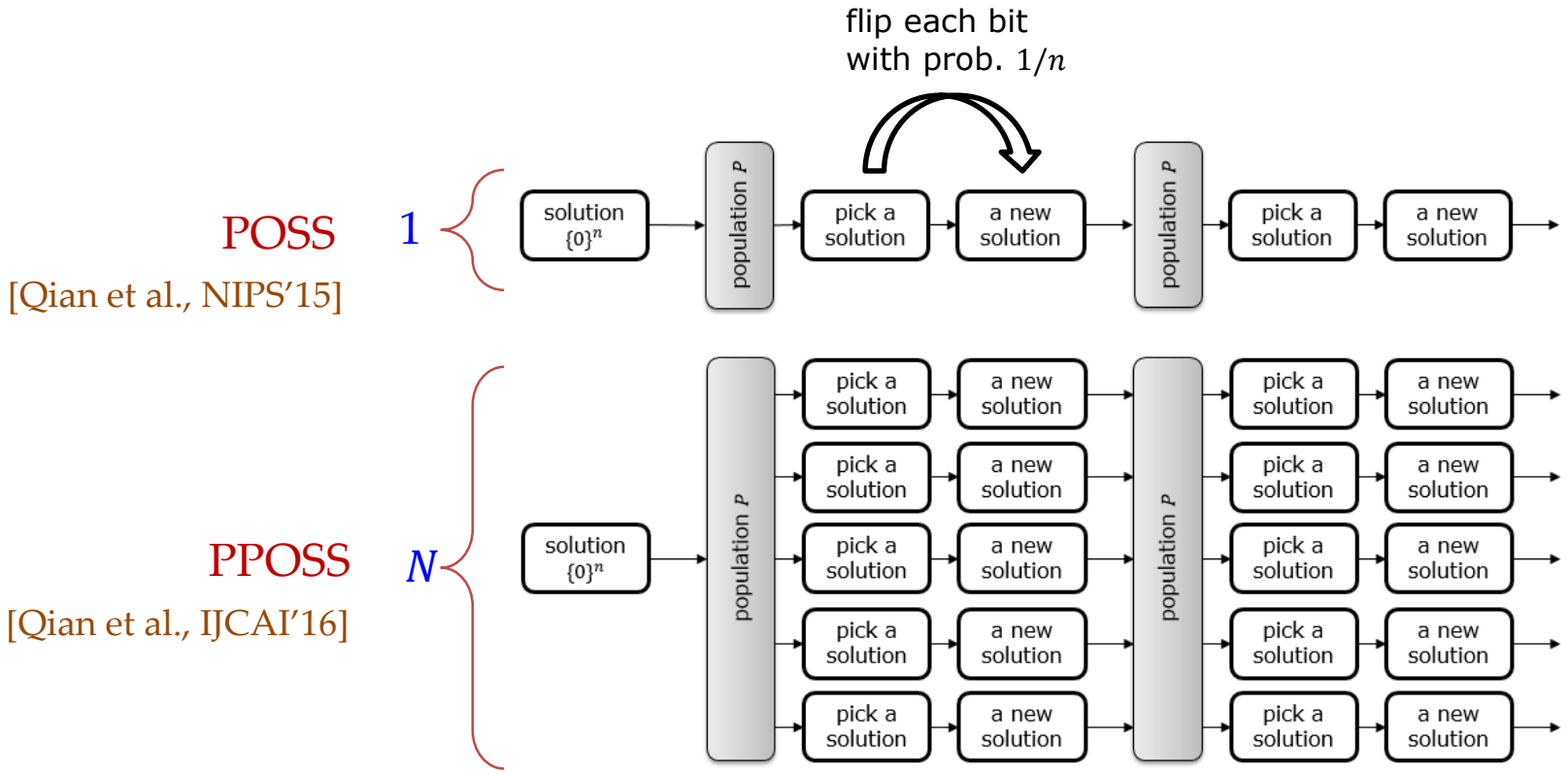
achieve nearly linear runtime speedup while keeping the solution quality

Require centralized access to the whole data set

restrict the application to large-scale real-world problems

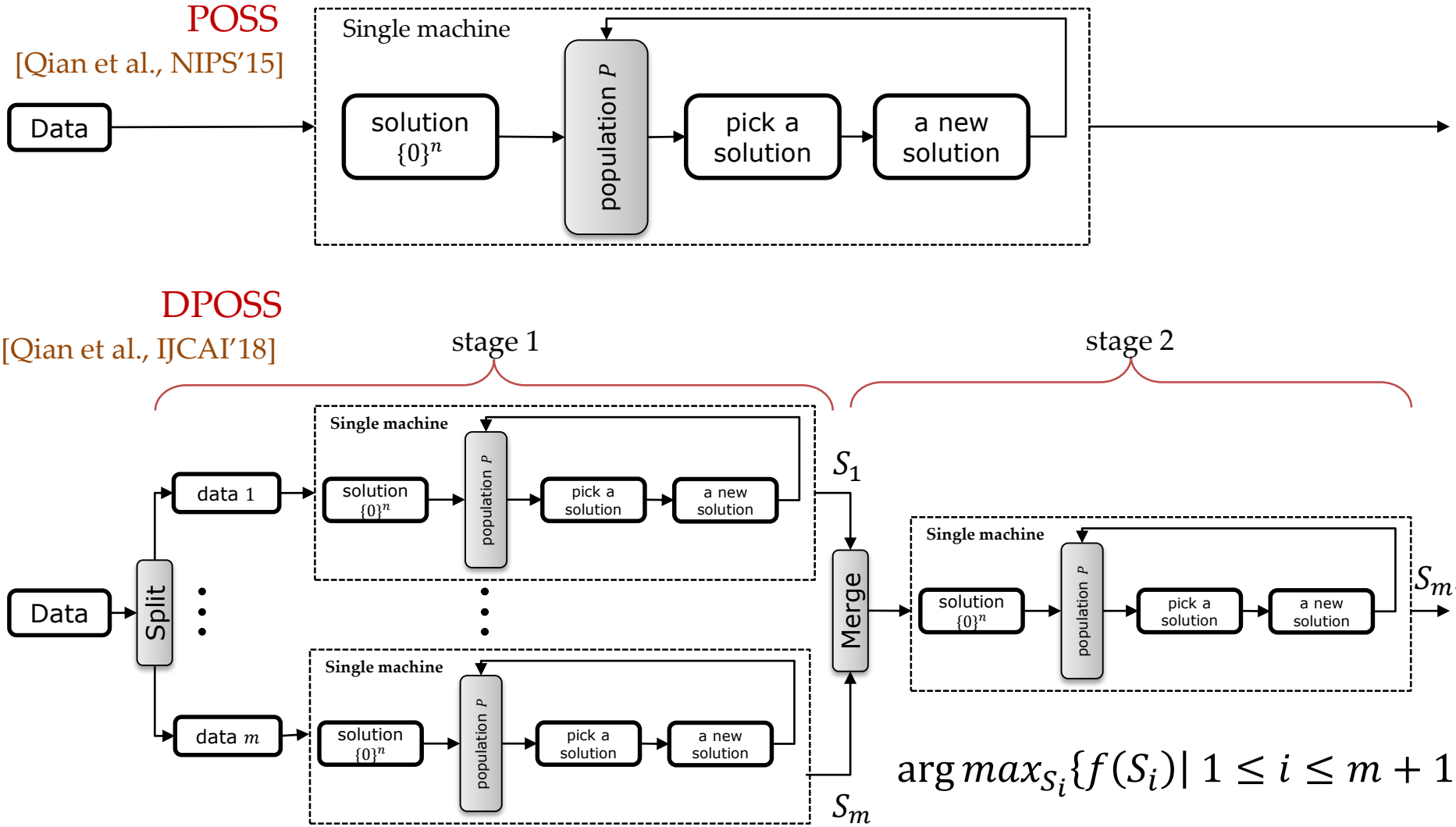
Can we make the Pareto optimization method distributable?

Distributed Pareto optimization for subset selection



Require centralized access to the whole data set

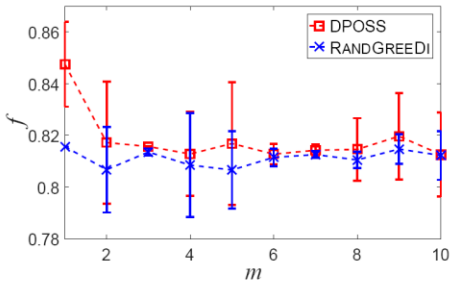
Distributed Pareto optimization for subset selection



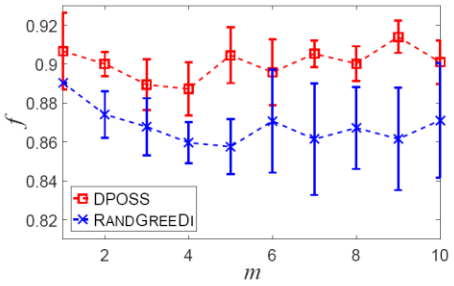
Experiments on sparse regression

Compare DPOSS with the state-of-the-art distributed greedy algorithm RandGreeDi [Mirzasoleiman et al., JMLR'16] under different number of machines

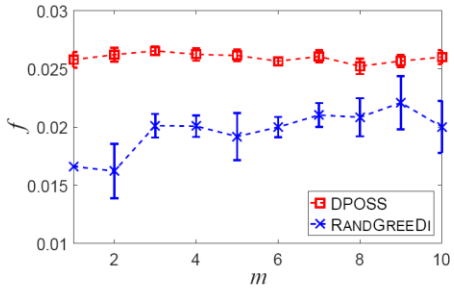
On regular-scale data sets



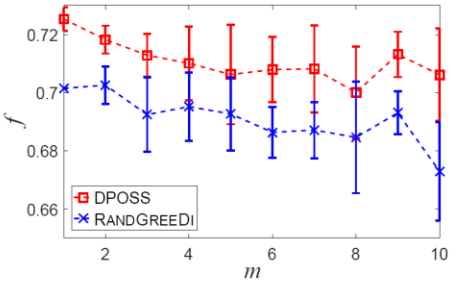
(a) *MicroMass* $(n = 1,300)$



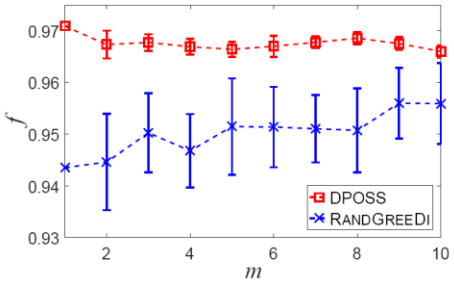
(b) *colon-cancer* $(n = 2,000)$



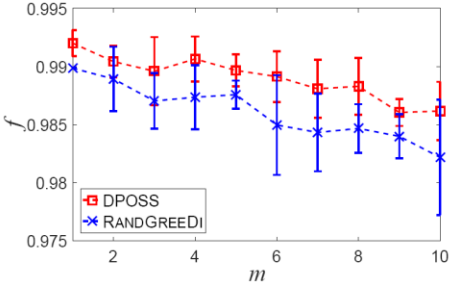
(c) *SVHN* $(n = 3,072)$



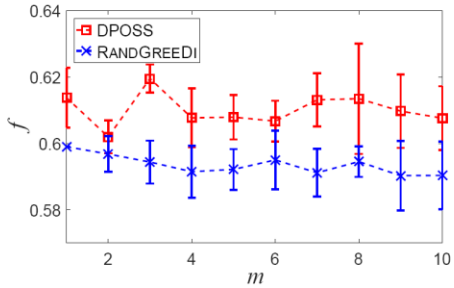
(d) *gisette* $(n = 5,000)$



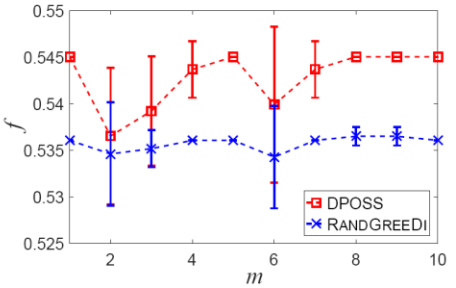
(e) *GHG-Network* $(n = 5,232)$



(f) *leukemia* $(n = 7,129)$



(g) *Arcene* $(n = 10,000)$



(h) *Dexter* $(n = 20,000)$

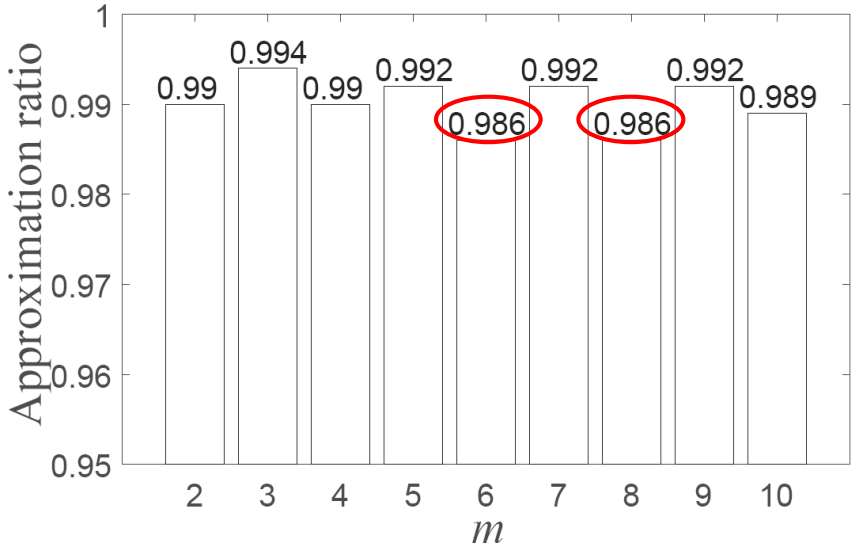
DPOSS is always better than RandGreeDi

Experiments on sparse regression

On regular-scale data sets

$$ratio = \frac{DPOSS}{POSS}$$

DPOSS is very close to the centralized POSS



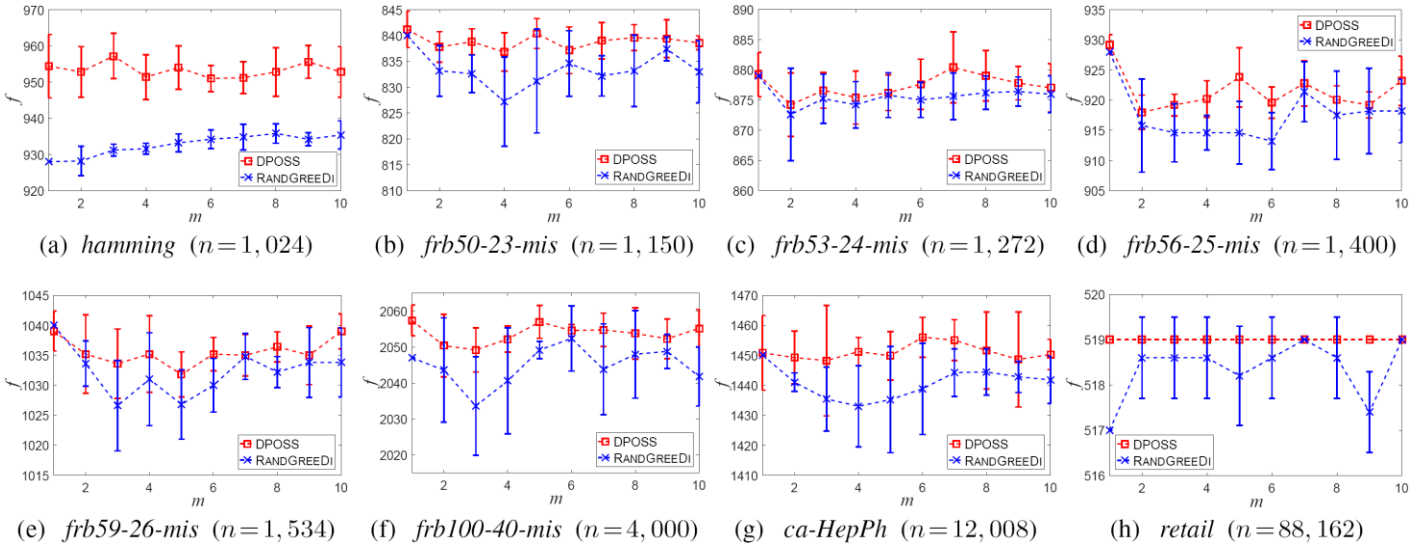
On large-scale data sets

DPOSS is better than RandGreeDi

Data set	DPOSS	RANDGREEDI
Gas-sensor-flow ($n = 120,432$)	.818±.005	.710±.017
Twin-gas-sensor ($n = 480,000$)	.601±.014	.470±.025
Gas-sensor-sample ($n = 1,950,000$)	.289±.029	.245±.018

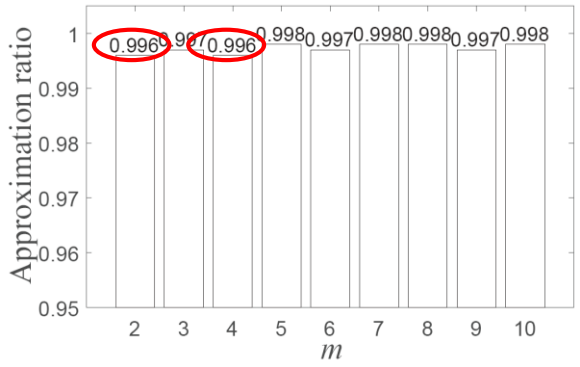
Experiments on maximum coverage

On regular-scale data sets



(a) *hamming* $(n = 1,024)$ (b) *frb50-23-mis* $(n = 1,150)$ (c) *frb53-24-mis* $(n = 1,272)$ (d) *frb56-25-mis* $(n = 1,400)$

(e) *frb59-26-mis* $(n = 1,534)$ (f) *frb100-40-mis* $(n = 4,000)$ (g) *ca-HepPh* $(n = 12,008)$ (h) *retail* $(n = 88,162)$



On large-scale data sets

Data set	DPOSS	RANDGREEDI
accident $(n = 340,183)$	175 ± 1	170.6 ± 1.34
kosarak $(n = 990,002)$	9263 ± 0	9263 ± 0

DPOSS is very close to the centralized POSS, and is better than RandGreeDi

Pareto optimization for subset selection

achieve superior performance on diverse variants of subset selection both theoretically and empirically

Parallel Pareto optimization for subset selection

achieve nearly linear runtime speedup while keeping the solution quality

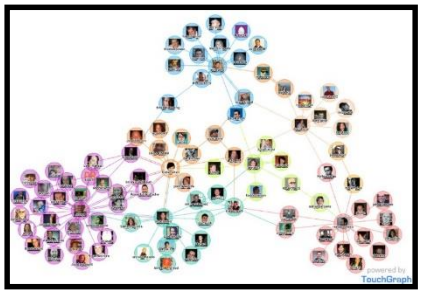
Distributed Pareto optimization for subset selection

achieve very close performance to the centralized algorithm

Noise

Previous analyses often assume that the exact value of the objective function can be accessed

However, in many applications of subset selection, only a noisy value of the objective function can be obtained



Influence maximization

The objective function $f(X)$: the expected number of nodes activated by propagating from X

noise

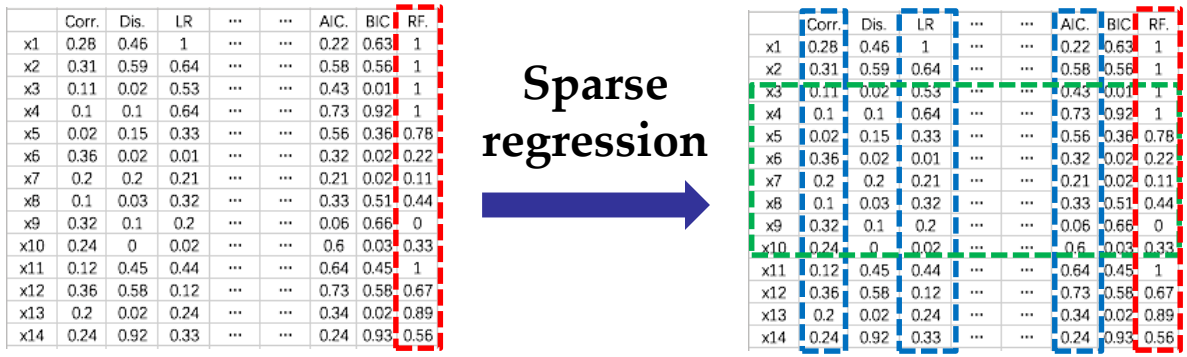
The average number of active nodes of independent diffusion processes

[Kempe et al., KDD'03]

Noise

Previous analyses often assume that the exact value of the objective function can be accessed

However, in many applications of subset selection, only a noisy value of the objective function can be obtained



How about the performance for noisy subset selection?

Outline

- Introduction
- Pareto optimization for subset selection
- Pareto optimization for large-scale subset selection
- **Pareto optimization for noisy subset selection**
- Conclusion

Noisy subset selection

Subset selection: given all items $V = \{v_1, \ldots, v_n\}$, an objective function $f: 2^V \rightarrow \mathrm{R}$ and a budget B, it is to find a subset $X \subseteq V$ such that

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

Noise

- Multiplicative: $(1-\epsilon)f(X) \leq F(X) \leq (1+\epsilon)f(X)$
- Additive: $f(X) - \epsilon \leq F(X) \leq f(X) + \epsilon$

Applications: influence maximization, sparse regression

maximizing information gain in graphical models [Chen et al., COLT'15]

crowdsourced image collection summarization [Singla et al., AAAI'16]

Theoretical analysis for greedy algorithms

[Qian, Shi, Yu, Tang and Zhou, NIPS'17]

Multiplicative noise:

$\varepsilon \leq 1/B$ for a constant approximation ratio

$$f(X) \geq \frac{1}{1+\frac{2\epsilon B}{(1-\epsilon)\gamma}}\left(1-\left(\frac{1-\epsilon}{1+\epsilon}\right)^B\left(1-\frac{\gamma}{B}\right)^B\right)\cdot OPT$$

γ ⟶ submodularity ratio

Additive noise:

$$f(X) \geq \left(1-\left(1-\frac{\gamma}{B}\right)^B\right)\cdot OPT - \left(\frac{2B}{\gamma}-\frac{2B}{\gamma}e^{-\gamma}\right)\epsilon$$

The noiseless approximation guarantee [Das & Kempe, ICML'11]

$$f(X) \geq \left(1-\left(1-\frac{\gamma}{B}\right)^B\right)\cdot OPT \geq (1-e^{-\gamma})\cdot OPT$$

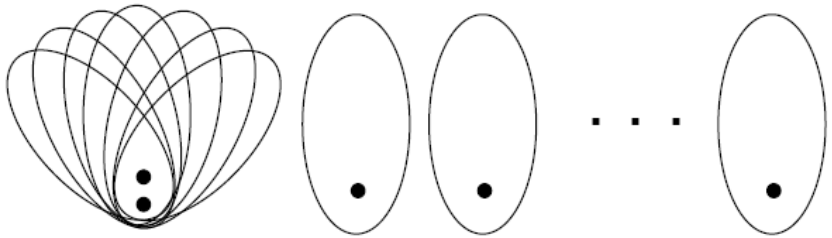
a constant approximation ratio

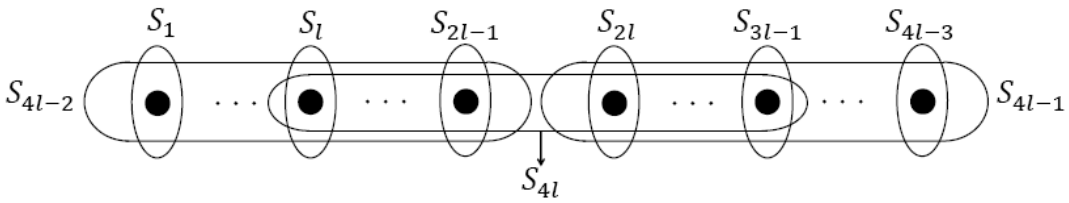
The performance largely degrades in noisy environments

Theoretical analysis for POSS

- POSS can generally achieve the same approximation guarantee in both multiplicative and additive noises
- POSS has a better ability of avoiding the misleading search direction led by noise

Maximum coverage





Greedy: very bad approximation
[Hassidim & Singer, COLT'17]

POSS: find the optimal solution through multi-bit search

POSS: find the optimal solution through backward search

PONSS

In our previous work, threshold selection was theoretically shown to be tolerant to noise [Qian et al., ECJ'18]

$$f(X) \geq f(Y) \longrightarrow f(X) \geq f(Y) + \theta$$

Exponentially decrease the running time

POSS

"better"
$$X \preccurlyeq Y \Leftrightarrow \begin{cases} f(X) \geq f(Y) \\ |X| \leq |Y| \end{cases}$$

Conservative

PONSS [Qian et al., NIPS'17]

Multiplicative:
$$X \preccurlyeq Y \Leftrightarrow \begin{cases} f(X) \geq \dfrac{1+\theta}{1-\theta} f(Y) \\ |X| \leq |Y| \end{cases}$$

Additive:
$$X \preccurlyeq Y \Leftrightarrow \begin{cases} f(X) \geq f(Y) + 2\theta \\ |X| \leq |Y| \end{cases}$$

Theoretical analysis

Multiplicative noise:

$\gamma = 1$ (submodular), ϵ is a constant

PONSS ($\theta \geq \epsilon$)

$$f(X) \geq \frac{1-\epsilon}{1+\epsilon}\left(1-\left(1-\frac{\gamma}{B}\right)^B\right) \cdot OPT$$

a constant approximation ratio

IV significantly better

POSS & Greedy

$$f(X) \geq \frac{1}{1+\frac{2\epsilon B}{(1-\epsilon)\gamma}}\left(1-\left(\frac{1-\epsilon}{1+\epsilon}\right)^B\left(1-\frac{\gamma}{B}\right)^B\right) \cdot OPT$$

$\Theta(1/B)$ approximation ratio

Additive noise:

PONSS ($\theta \geq \epsilon$)

$$f(X) \geq \left(1-\left(1-\frac{\gamma}{B}\right)^B\right) \cdot OPT - 2\epsilon$$

IV significantly better

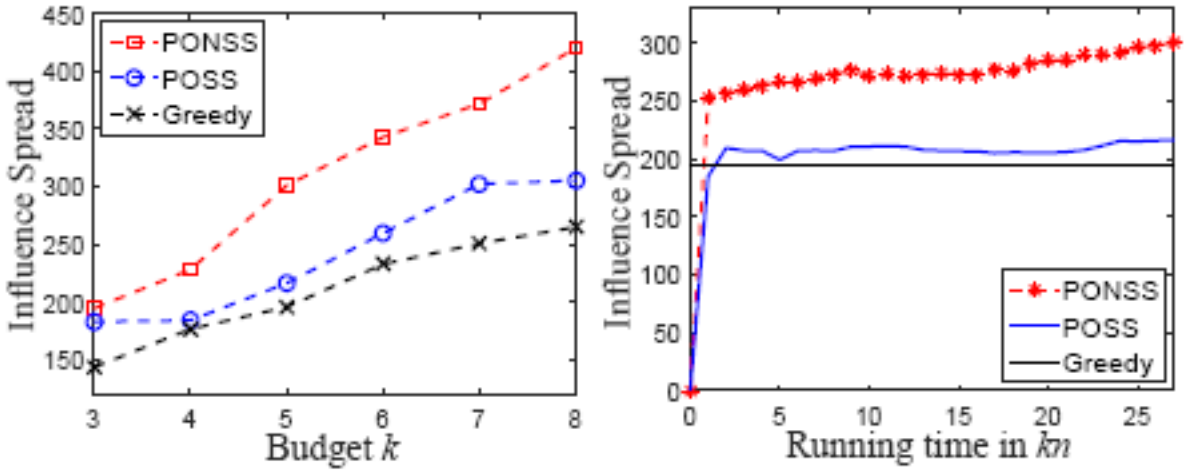
POSS & Greedy

$$f(X) \geq \left(1-\left(1-\frac{\gamma}{B}\right)^B\right) \cdot OPT - \left(\frac{2B}{\gamma} - \frac{2B}{\gamma}e^{-\gamma}\right)\epsilon$$

Experimental results - influence maximization

PONSS (red line) vs POSS (blue line) vs Greedy (black line):

- Noisy evaluation: the average of 10 independent Monte Carlo simulations
- The output solution: the average of 10,000 independent Monte Carlo simulations

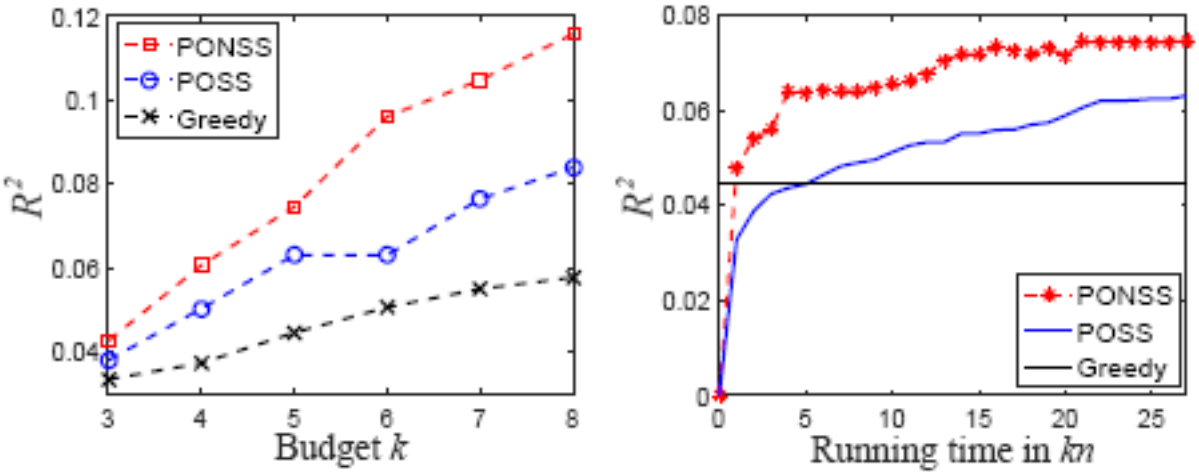


(b) *Weibo* (10,000 #nodes, 162,371 #edges)

Experimental results - sparse regression

PONSS (red line) vs POSS (blue line) vs Greedy (black line):

- Noisy evaluation: a random sample of 1,000 instances
- The output solution: the whole data set



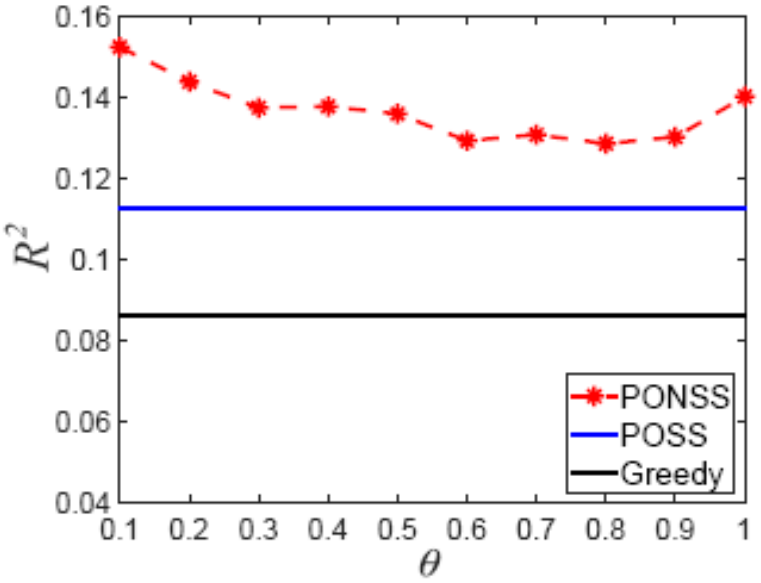
(a) *protein* (24,387 #inst, 357 #feat)

Experimental results – sensitivity to θ

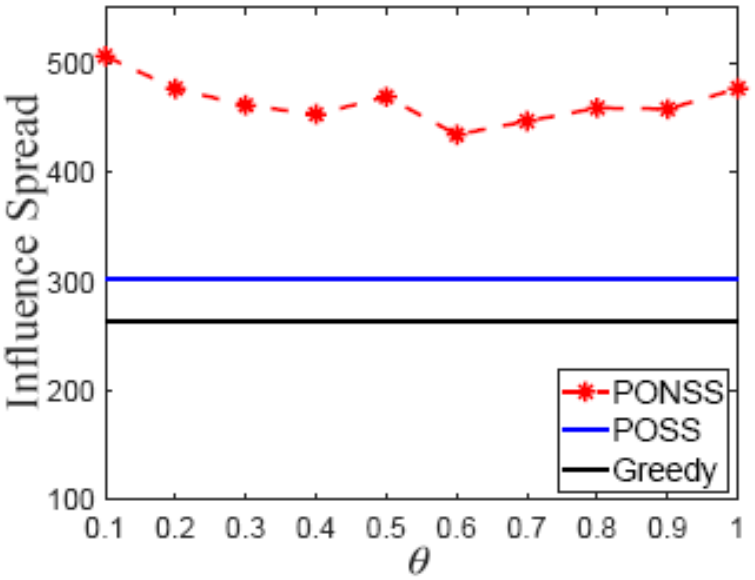
PONSS (red line) vs POSS (blue line) vs Greedy (black line):

- $\theta = \{0.1, 0.2, \ldots, 1\}$

The performance of PONSS is not sensitive to θ

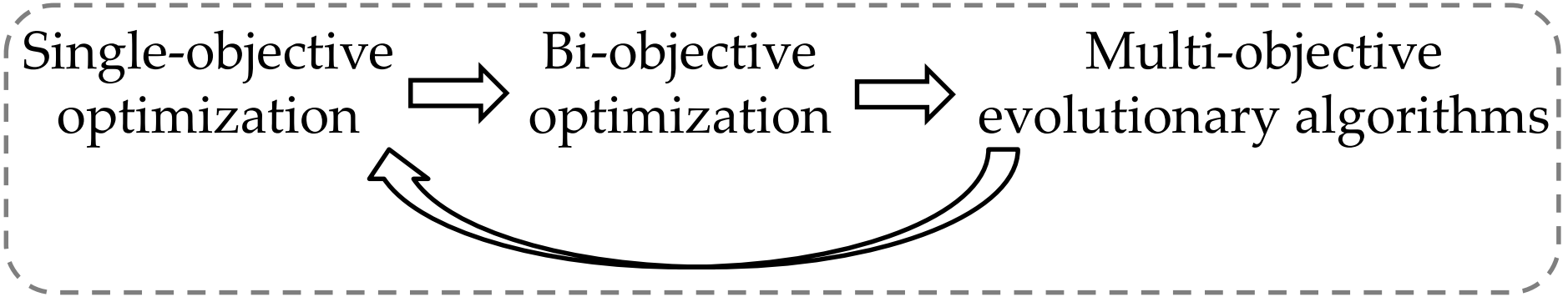


(a) *protein*



(b) *Weibo*

Conclusion



- Pareto optimization for subset selection
  - Show superior performances theoretically and empirically
- Pareto optimization for large-scale subset selection
  - Introduce parallel and distributed strategies
- Pareto optimization for noisy subset selection
  - Introduce noise-aware strategies

Future work

- Problem issues
  - Non-monotone objective functions [Qian et al., 2017]
  - Continuous submodular objective functions
  - Other than subset selection [Neumann & Wegener, 2006; Friedrich et al., 2010; Neumann et al., 2011; Qian et al., 2015]
- Algorithm issues
  - More complicated MOEAs
- Theory issues
  - Beat the best known approximation guarantee
- Application issues
  - Attempts on more large-scale real-world applications

For details

- F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 2006, 5(3): 305-319.
- 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.
- F. Neumann, J. Reichel and M. Skutella. Computing minimum cuts by randomized search heuristics. *Algorithmica*, 2011, 59(3): 323-342.
- T. Friedrich and F. Neumann. Maximizing submodular functions under Matroid constraints by evolutionary algorithms. *Evolutionary Computation*, 2015, 23(4): 543-558.
- C. Qian, Y. Yu and Z.-H. Zhou. Pareto ensemble pruning. In: *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*, Austin, TX, 2015.
- 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)*, Buenos Aires, Argentina, 2015.

For details

- C. Qian, Y. Yu and Z.-H. Zhou. Subset selection by Pareto optimization. In: *Advances in Neural Information Processing Systems 28 (NIPS'15)*, Montreal, Canada, 2015.

- C. Qian, J.-C. Shi, Y. Yu, K. Tang and Z.-H. Zhou. Parallel Pareto optimization for subset selection. In: *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, New York, NY, 2016.

- C. Qian, J.-C. Shi, Y. Yu and K. Tang. On subset selection with general cost constraints. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, Melbourne, Australia, 2017.

- C. Qian, J.-C. Shi, Y. Yu, K. Tang and Z.-H. Zhou. Optimizing ratio of monotone set functions. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, Melbourne, Australia, 2017.

- C. Qian, J.-C. Shi, K. Tang and Z.-H. Zhou. Constrained monotone k-submodular function maximization using multi-objective evolutionary algorithms with theoretical guarantee. *IEEE Transactions on Evolutionary Computation*, in press.

For details

- C. Qian, J.-C. Shi, Y. Yu, K. Tang and Z.-H. Zhou. Subset selection under noise. In: *Advances in Neural Information Processing Systems 30 (NIPS'17)*, Long Beach, CA, 2017.
- C. Qian, Y. Yu, K. Tang, X. Yao and Z.-H. Zhou. Maximizing non-monotone/non-submodular functions by multi-objective evolutionary algorithms. *CORR abs/1711.07214*, 2017.
- C. Qian, Y. Zhang, K. Tang and X. Yao. On multiset selection with size constraints. In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*, New Orleans, LA, 2018.
- C. Qian, G. Li, C. Feng and K. Tang. Distributed Pareto optimization for subset selection. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, Stockholm, Sweden, 2018.
- C. Qian, C. Feng and K. Tang. Sequence selection by Pareto optimization. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, Stockholm, Sweden, 2018.
- C. Qian, Y. Yu and Z.-H. Zhou. Analyzing evolutionary optimization in noisy environments. *Evolutionary Computation*, 2018, 26(1): 1-41.

Codes available at http://staff.ustc.edu.cn/~chaoqian/

THANK YOU !