LAMDA
Learning And Mining from DatA
http://www.lamda.nju.edu.cn

PPSN 2020
Leiden, The Netherlands

Pareto Optimization for Subset Selection: Theories and Practical Algorithms

Chao Qian and Yang Yu

LAMDA Group, Nanjing University, China

Email: {qianc, yuy}@lamda.nju.edu.cn

Outline

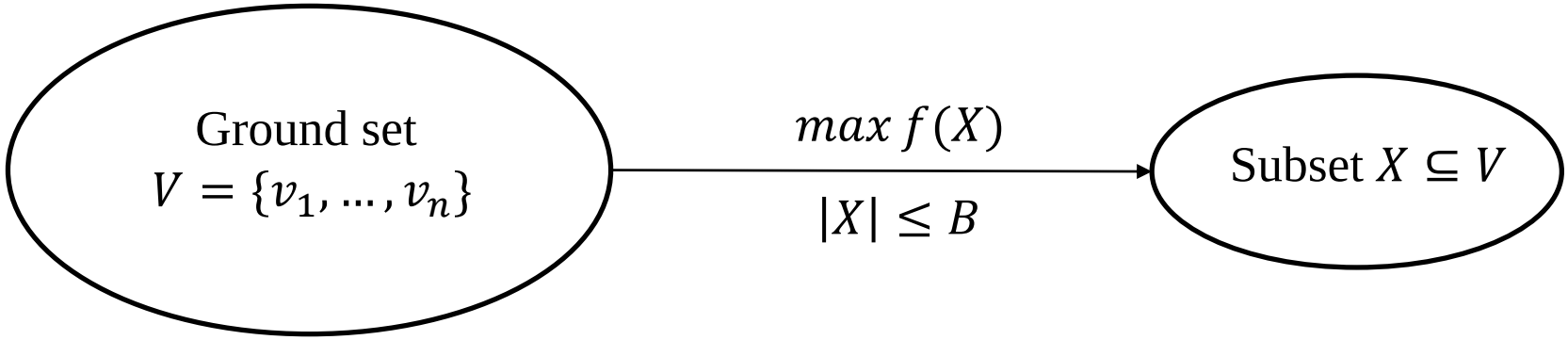
- **Introduction**
- Pareto optimization for subset selection
- Pareto optimization for large-scale subset selection
- Pareto optimization for noisy subset selection
- Pareto optimization for dynamic subset selection
- Conclusion

Subset selection

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

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

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

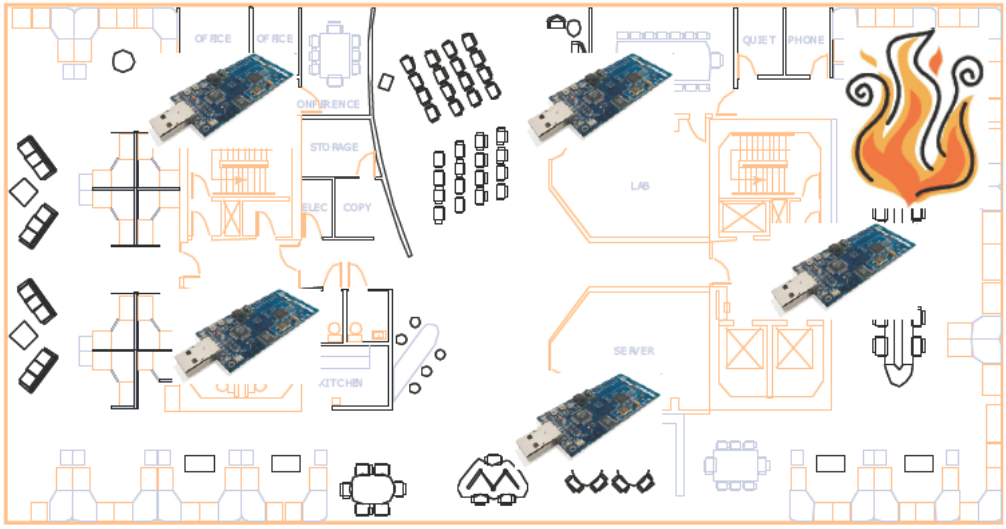


Subset selection has diverse applications, which have different meanings on the item v_i and the objective f

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



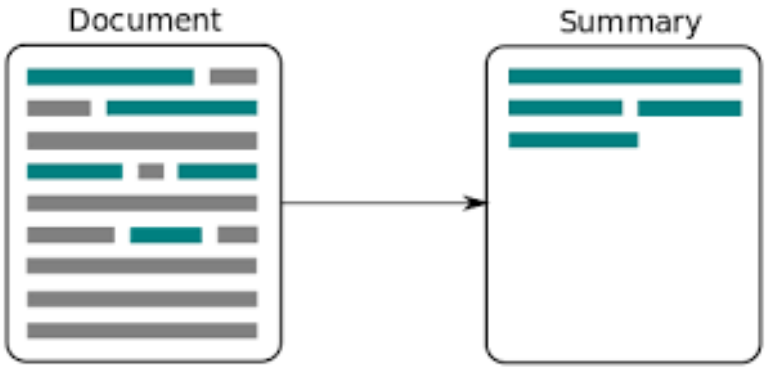
Fire detection

Item v_i: a place to install a sensor

Objective f: entropy

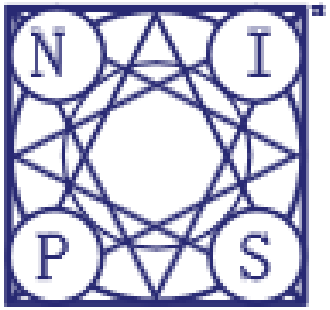
Application - document summarization

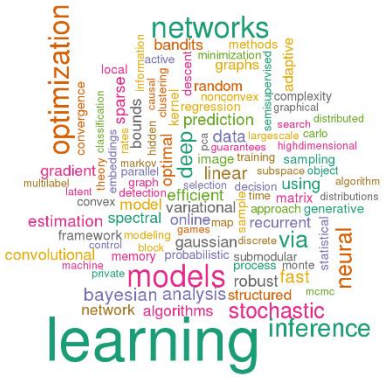
Document summarization [Lin & Bilmes, ACL'11]: select a few sentences to best summarize the documents



Item v_i: a sentence

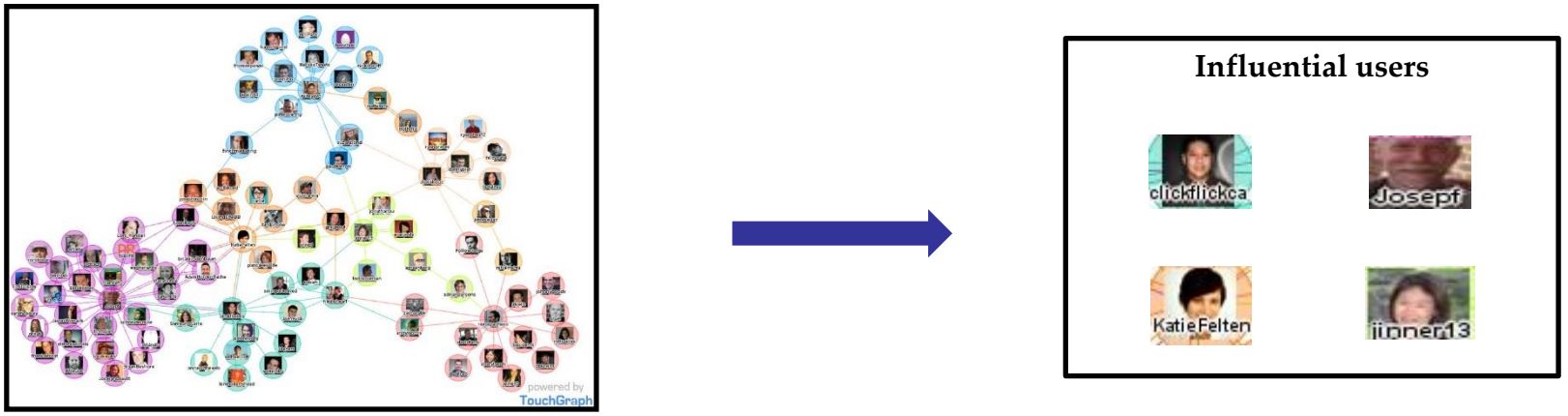
Objective f: summary quality





Application - influence maximization

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

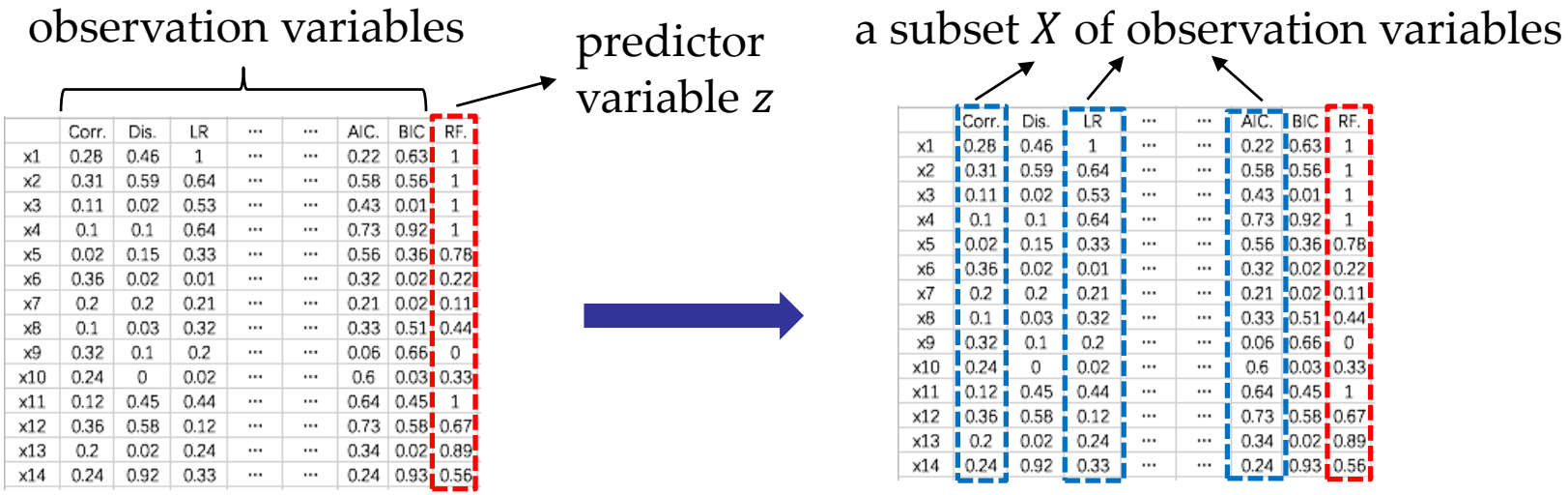


Item v_i: a social network user

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

Application - sparse regression

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

Item v_i: an observation variable

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

variance

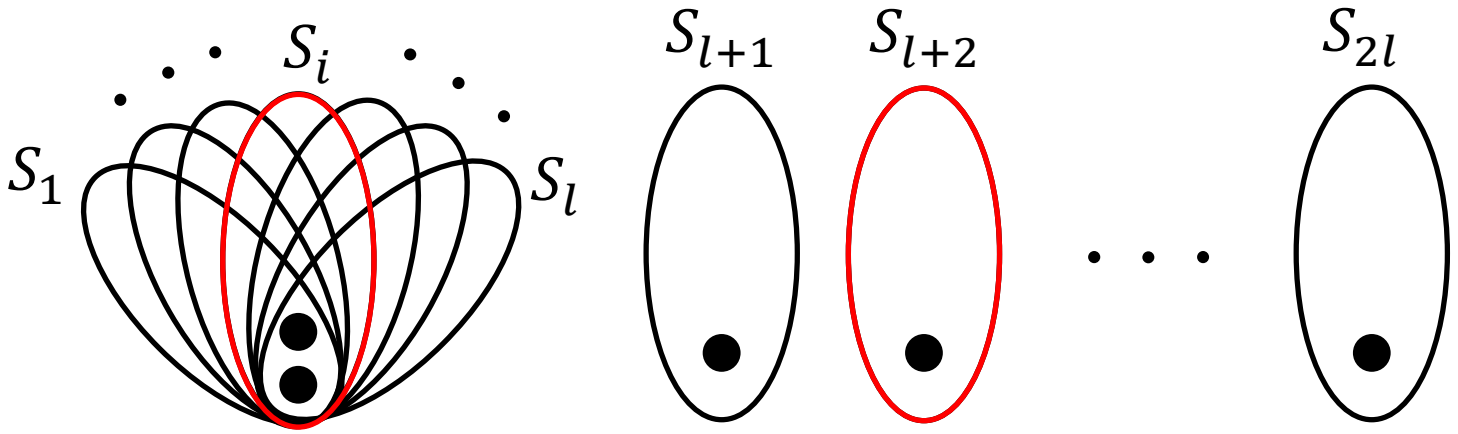
mean squared error

Application - maximum coverage

Maximum coverage [Feige, JACM'98]: select at most B sets from n given sets $V = \{S_1, \ldots, S_n\}$ to make the size of their union maximal

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

Example: $\forall i \leq l$, S_i contains the same two elements, $\forall i > l$, S_i contains one unique element; $n = 2l$, $B = 2$



Item v_i: a set of elements

Objective f: size of the union

Subset selection

Subset selection is to select a subset of size at most 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, 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
sensor placement	a place to install a sensor	entropy
document sum		summary quality
influence max		influence spread
sparse regression	an observation variable	squared multiple correlation
maximum coverage	a set of elements	size of the union

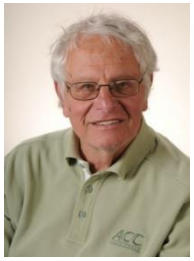
Many applications, but NP-hard in general!

Subset selection

Subset selection is to select a subset of size at most 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, to find a subset $X \subseteq V$ such that

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



George Nemhauser

John Von Neumann Theory Prize

[Mathematical Programming 1978]

f: monotone and submodular

Greedy algorithm: $(1 - 1/e)$-approximation

Application to influence maximization

KDD Test of Time Award

[Kempe et al., KDD'03]

Extension to non-submodular

ICML Best Paper

[Das & Kempe, ICML'11]

ICML&NIPS Best Paper

[Iyer, et al., ICML'13]

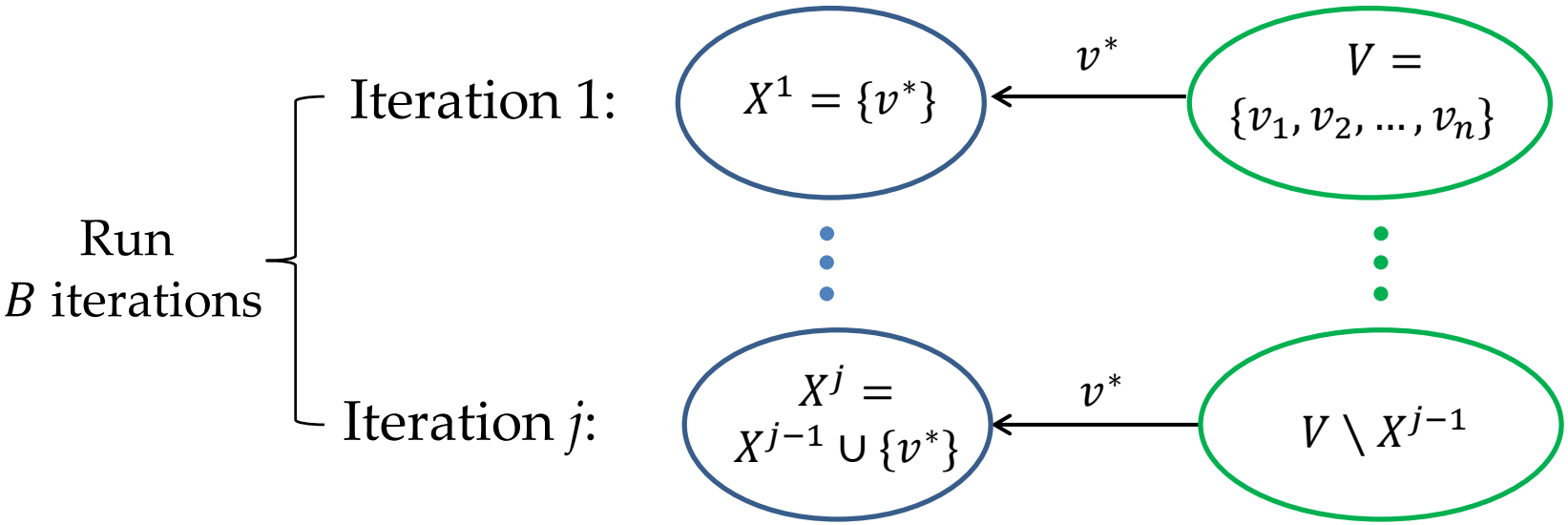
[Iyer & Bilmes, NIPS'13]

Previous approaches

- Greedy algorithm

Process: iteratively select one item maximizing the increment on f

$$v^* = \arg max_{v \in V \setminus X^{j-1}} f\left(X^{j-1} \cup \{v\}\right) - f\left(X^{j-1}\right)$$



X^j: the subset obtained after j iterations

Approximation guarantees

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, to find a subset $X \subseteq V$ such that

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

f: monotone and submodular

The approximation guarantee [Nemhauser et al., MP'78]:

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

The subset X output by the greedy algorithm satisfies

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

OPT: the optimal function value

Monotone and submodular

A set function $f\colon 2^V \to \mathrm{R}$ requires a solution to be a subset of V

Monotone: the function value increases as a set extends, i.e.,

$$\forall X \subseteq Y \subseteq V\colon f(X) \leq f(Y)$$

Submodular [Nemhauser et al., MP'78] : satisfy the natural diminishing returns property, i.e.,

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

or equivalently,

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

or equivalently,

$$\forall X, Y \subseteq V\colon f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$$

Submodular applications

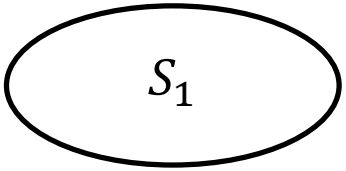
Maximum coverage [Feige, JACM'98] : select at most B sets from n given sets $V = \{S_1, \dots, S_n\}$ to make the size of their union maximal

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

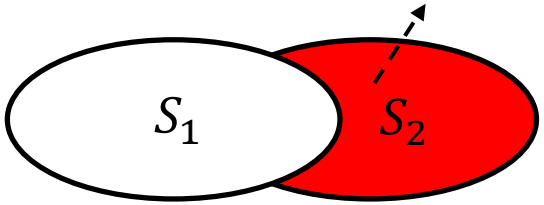
Monotone: $\forall X \subseteq Y \subseteq V: f(X) \leq f(Y)$

$f(Y) - f(X)$

$X = \{S_1\}$



$Y = \{S_1, S_2\}$

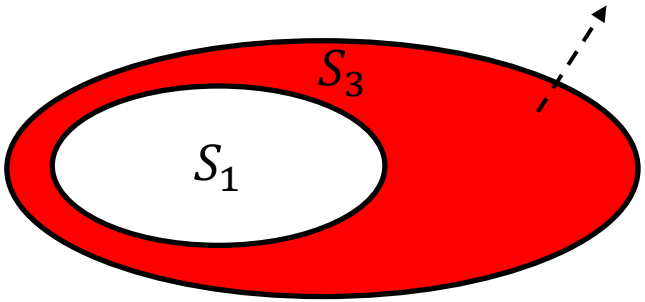


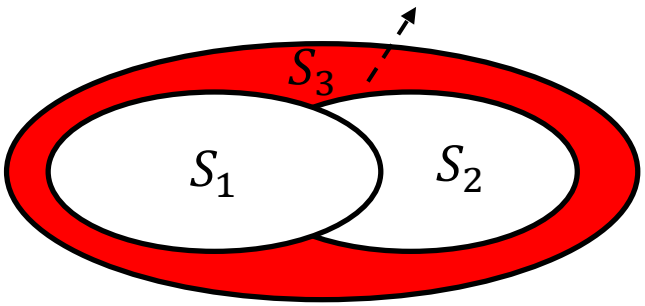
Submodular: $\forall X \subseteq Y \subseteq V, v \notin Y: f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$

$X = \{S_1\}$

$Y = \{S_1, S_2\}$

$v = S_3$





Submodular applications

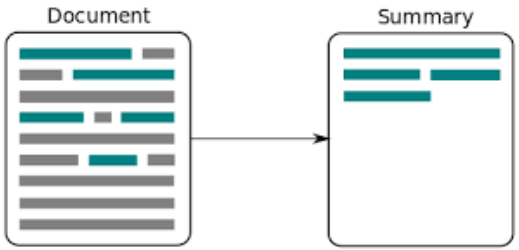
Maximum coverage [Feige, JACM'98] : select at most B sets from n given sets $V = \{S_1, \ldots, S_n\}$ to make the size of their union maximal

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

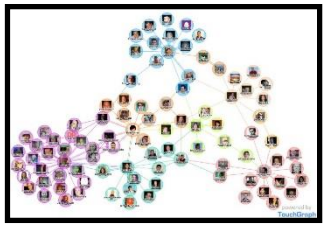
More applications:

- Sensor placement

- Document summarization



- Influence maximization



Their objective functions are all monotone and submodular

Approximation guarantees

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, to find a subset $X \subseteq V$ such that

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

f: monotone and submodular

↳ The approximation guarantee [Nemhauser et al., MP'78] :

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

f: monotone

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

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

Submodular ratio γ: to what extent f satisfies the submodular property

Submodular ratio

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

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

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

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

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

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

Characterize to what extent a set function f satisfies the submodular property

For example, when f is monotone,

- $\forall U, k: \gamma_{U,k}(f) \in [0,1]$, the larger, more close to submodular
- f is submodular if and only if $\forall U, k: \gamma_{U,k}(f) = 1$

Non-submodular applications

Submodular ratio [Das & Kempe, ICML'11; Zhang & Vorobeychi, AAAI'16] : characterize to what extent a general set function satisfies the submodular property

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

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

Lower bounds on submodular ratio for some non-submodular applications

- Sparse regression: $\gamma_{U,k}(f) \geq \lambda_{min}(\mathrm{C}, |U| + k)$ [Das & Kempe, ICML'11]
- Sparse support selection: $\gamma_{U,k}(f) \geq m/M$ [Elenberg et al., Annals of Statistics'18]
- Bayesian experimental design [Bian et al., ICML'17]:

$$\gamma_{U,k}(f) \geq \beta^2 / \left(\|\mathrm{V}\|^2 (\beta^2 + \sigma^{-2} \|\mathrm{V}\|^2) \right)$$

- Determinantal function maximization [Qian et al., IJCAI'18]:

$$\alpha_f \geq (\lambda_n(\mathrm{A}) - 1) / \left((\lambda_1(\mathrm{A}) - 1) \prod_{i=1}^{n-1} \lambda_i(\mathrm{A}) \right)$$

Approximation guarantees

f: monotone and submodular

Optimal [Nemhauser & Wolsey, MOR'78]

↳ The approximation guarantee [Nemhauser et al., MP'78]:

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

f: monotone

Optimal [Harshaw et al., ICML'19]

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

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

Good optimization performance:

- Good approximation guarantee, i.e., good performance in worst cases ✓
- Practical performance is much better (e.g., close to optima) in most cases ?

The greedy nature

Previous approaches (con't)

- Relaxation method

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

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

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

$$\Updownarrow$$

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

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

Variants of subset selection

- Subset selection

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

$1 - 1/e^{\gamma}$ [Das & Kempe, ICML'11]

- General constraints

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

$(\alpha/2)\,(1 - 1/e^{\alpha})$ [Zhang & Vorobeychik, AAAI'16]

- Multiset selection

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]

- k-subsets selection

X: a subset → k subsets

$1/2$ [Ohsaka & Yoshida, NIPS'15]

- Sequence selection

X: a subset → a sequence

$1 - e^{-1/(2\Delta)}$ [Tschiatschek et al, AAAI'17]

- Ratio optimization

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

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

[Bai et al., ICML'16]

Motivation

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

Two conflicting objectives:

1. Optimize the objective f $\quad max_{X \subseteq V} \ f(X)$
2. Keep the size small $\quad min_{X \subseteq V} \ max\{|X| - B, 0\}$

Previous theoretical studies have disclosed the advantage of solving single-objective constrained optimization by MOEAs

[Neumann & Wegener, NC'06; Friedrich et al., ECJ'10; Neumann et al., Algorithmica'11; Yu et al., AIJ'12]

Why not optimize the bi-objective formulation?

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

Outline

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

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

Pareto optimization

The basic idea:

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

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

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

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

subset selection
and some variants

Bi-objective optimization

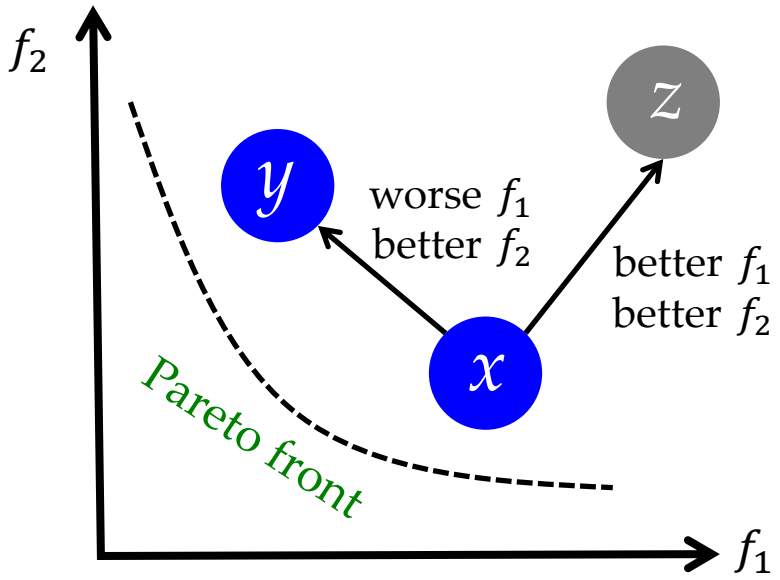
$min_{\boldsymbol{x}}\ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$

$\boldsymbol{x}$ dominates $\boldsymbol{z}$:

$f_1(\boldsymbol{x}) < f_1(\boldsymbol{z}) \wedge f_2(\boldsymbol{x}) < f_2(\boldsymbol{z})$

$\boldsymbol{x}$ is incomparable with $\boldsymbol{y}$:

$f_1(\boldsymbol{x}) > f_1(\boldsymbol{y}) \wedge f_2(\boldsymbol{x}) < f_2(\boldsymbol{y})$



Pareto optimization

The basic idea:

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

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

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

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

How to transform?

⇨

Bi-objective optimization

$min_{\boldsymbol{x}}\ (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$

⇩

Multi-objective evolutionary algorithms

⇦

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

↘

Different from traditional multi-objective optimization

Subset selection with monotone submodular f

[Friedrich & Neumann, ECJ'15]

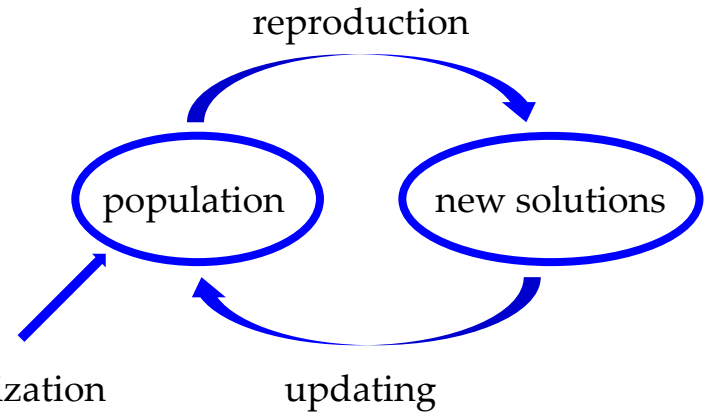
Exclude solutions with size larger than B

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

Transformation: ⇩

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

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 $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x} \in \{0,1\}^n$ 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

Subset selection with monotone f

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

Exclude solutions with size at least $2B$

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

Transformation: ⇩

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

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 $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x}$ 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

Subset selection with monotone f

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

Exclude solutions with size at least $2B$

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

Transformation: ⇩

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

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

Reproduction: pick a solution $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x}$ 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

- Selection: each solution in the population P is selected with probability $1/|P|$

  e.g., if P contains 10 solutions, each solution is selected with probability $1/10$

- Bit-wise mutation:

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

  e.g., the probability of flipping a specific bit of a solution is $(1/n)(1-1/n)^{n-1}$

Subset selection with monotone f

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

Exclude solutions with size at least $2B$

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

Transformation: ⇩

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

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

Reproduction: pick a solution $\boldsymbol{x}$ randomly from P, and flip each bit of $\boldsymbol{x}$ 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

- Selection: each solution in the population P is selected with probability $1/|P|$

  e.g., if P contains 10 solutions, each solution is selected with probability $1/10$

- Bit-wise mutation:

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

  e.g., the probability of flipping a specific bit of a solution is $(1/n)(1-1/n)^{n-1}$

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

Theoretical analysis

POSS can achieve the optimal approximation guarantee, previously obtained by the greedy algorithm

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

the expected number of iterations

the optimal 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

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

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

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

Main idea:

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

$i = 0$

initial solution $00 \ldots 0$

$|00 \ldots 0| = 0$

$f(00 \ldots 0) = 0$

$i = B$

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

let $m = B/\gamma$

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

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

Proof

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

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

X: a subset

Main idea:

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

$i = 0$? $i = B$

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 guarantee

Proof

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

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

a subset

Main idea:

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

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

Proof

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

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

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

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

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

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

Proof

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

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

X: a subset

Main idea:

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

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

$i \longrightarrow i + 1$ 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

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

X: a subset

Main idea:

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

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

- 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

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

Main idea:

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

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

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

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

Theoretical analysis

POSS can achieve the optimal approximation guarantee, previously obtained by the greedy algorithm

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

the optimal 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.

Experiments on sparse regression

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

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

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

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

observation variables

predictor variable z

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 - R^2 values

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

exhaustive search | greedy algorithms | relaxation methods

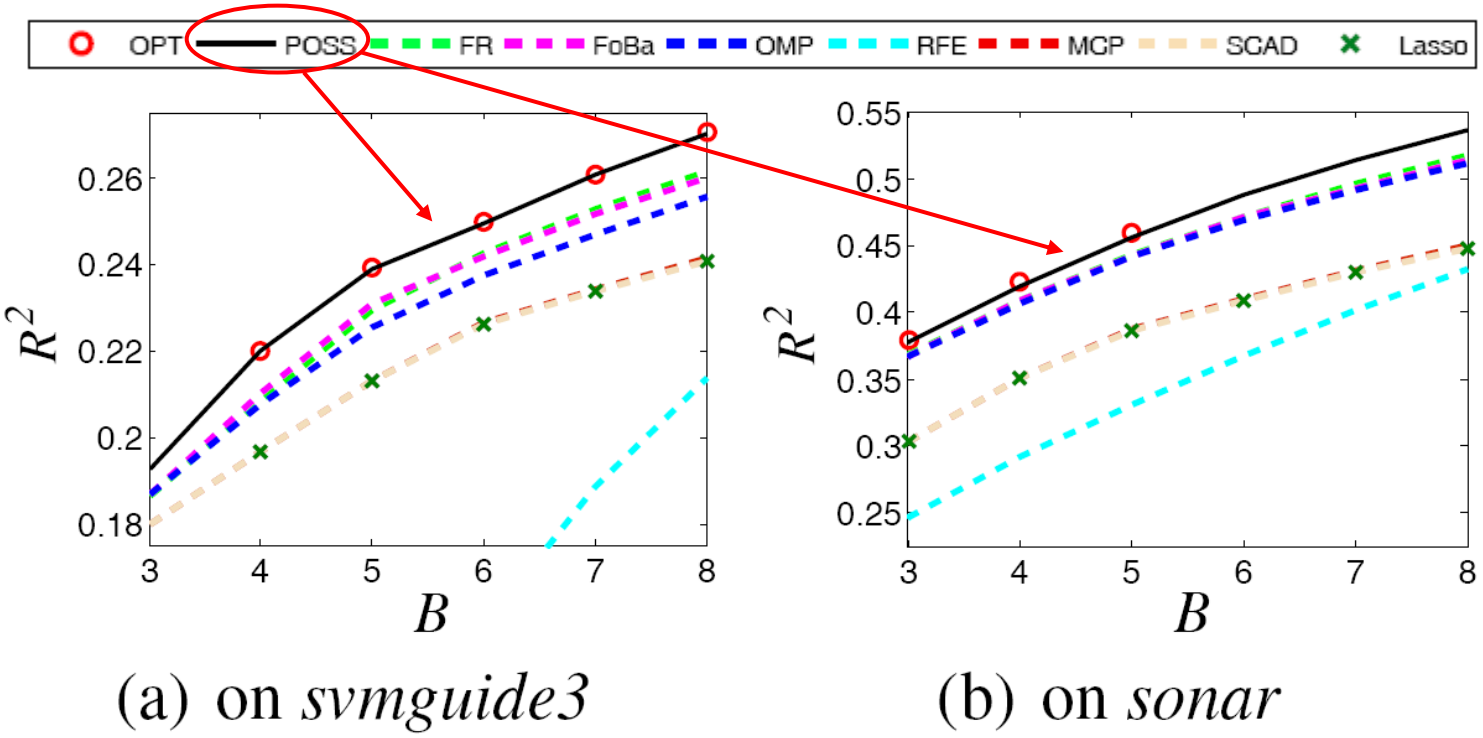
Data set	OPT	POSS	FR	FoBa	OMP	RFE	MCP
housing	.7437±.0297	.7437±.0297	.7429±.0300•	.7423±.0301•	.7415±.0300•	.7388±.0304•	.7354±.0297•
eunite2001	.8484±.0132	.8482±.0132	.8348±.0143•	.8442±.0144•	.8349±.0150•	.8424±.0153•	.8320±.0150•
svmguide3	.2705±.0255	.2701±.0257	.2615±.0260•	.2601±.0279•	.2557±.0270•	.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

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

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

Experimental results - R^2 values

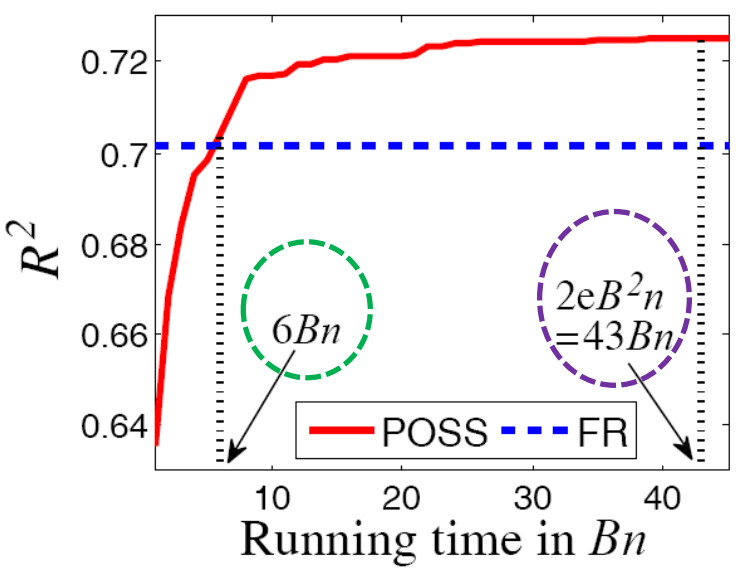
different size constraints: $B = 3 \rightarrow 8$



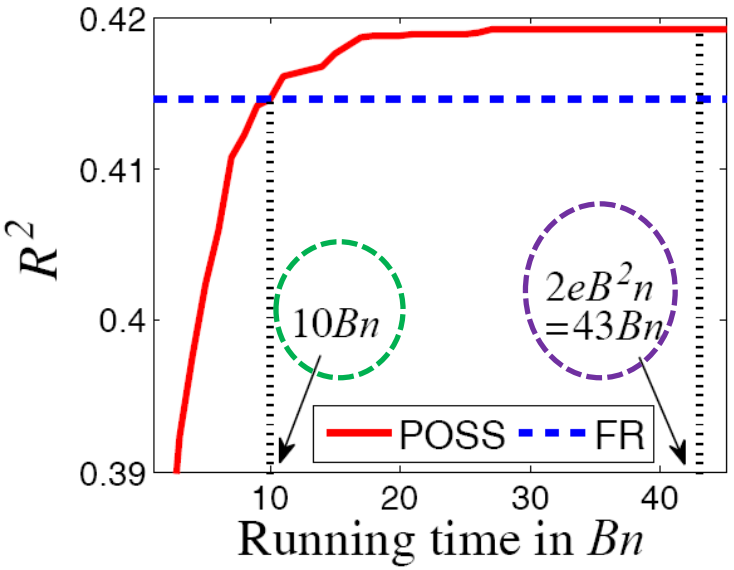
POSS tightly follows OPT, and has a clear advantage over the rest algorithms

Experimental results – running time

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



(a) on *gisette*



(b) on *farm-ads*

theoretical running time

POSS can be much more efficient in practice

Pareto optimization vs. Greedy algorithm

Greedy algorithm:

- Generate a new solution by adding a single item

  (i.e., single-bit forward search: $0 \rightarrow 1$)

- Keep only one solution

Pareto optimization:

- Generate a new solution by flipping each bit with prob. $1/n$
  - single-bit forward search : $0 \rightarrow 1$
  - backward search : $1 \rightarrow 0$
  - multi-bit search : $00 \rightarrow 11$
- Keep a set of non-dominated solutions due to bi-objective optimization

Pareto optimization may have a better ability of escaping from local optima

Variants of subset selection

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

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

- Multiset selection $max_{\boldsymbol{x}\in Z_+^n} f(\boldsymbol{x}) \quad s.t. \quad |\boldsymbol{x}| \leq B$

  x_i: the number of times that the item v_i appears

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

  x_i: the subset where the item v_i appears

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

  x: a sequence where the order of items influences f

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

Variants of subset selection

- Subset selection $max_{\boldsymbol{x}\in\{0,1\}^n} f(\boldsymbol{x}) \quad s.t. \quad |\boldsymbol{x}| \leq B$ [Friedrich & Neumann, ECJ'15; Qian et al., NIPS'15]

- General constraints $max_{\boldsymbol{x}\in\{0,1\}^n} f(\boldsymbol{x}) \quad s.t. \quad c(\boldsymbol{x}) \leq B$ [Qian et al., IJCAI'17a]

- Multiset selection $max_{\boldsymbol{x}\in\mathbb{Z}_+^n} f(\boldsymbol{x}) \quad s.t. \quad |\boldsymbol{x}| \leq B$ [Qian et al., AAAI'18]
- k-subsets selection $max_{\boldsymbol{x}\in\{0,1,\ldots,k\}^n} f(\boldsymbol{x}) \quad s.t. \quad |\boldsymbol{x}| \leq B$ [Qian et al., TEvC'18]
- Sequence selection $max_{x\in\mathcal{S}} f(x) \quad s.t. \quad |x| \leq B$ [Qian et al., IJCAI'18]

- Ratio optimization $min_{\boldsymbol{x}\in\{0,1\}^n} f(\boldsymbol{x})/g(\boldsymbol{x})$ [Qian et al., IJCAI'17b]

Pareto optimization can achieve the best-known polynomial-time approximation guarantee, and perform well in practice

Ratio optimization

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

Transformation:

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

⇩

$min_{\boldsymbol{x}\in\{0,1\}^n}\ (f(\boldsymbol{x}), -g(\boldsymbol{x}))$ 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 $x \in \{0,1\}^n$
Process:

1: Select x from $\{0,1\}^n$ uniformly at random.
2: Let $P = \{x\}$ and $t = 0$.
3: **while** $t < T$ **do**
4: Select x from P uniformly at random.
5: Generate x' by flipping each bit of x with prob. $1/n$.
6: **if** $\nexists z \in P$ such that $z \prec x'$ **then**
7: $P = (P \setminus \{z \in P \mid x' \preceq z\}) \cup \{x'\}$.
8: $Q = \{z \in P \mid |z| = |x'|\}$.
9: $z_1 = \arg\min_{z\in Q} f(z)$, $z_2 = \arg\max_{z\in Q} g(z)$, $z_3 = \arg\min_{z\in Q} f(z)/g(z)$.
10: $P = (P \setminus Q) \cup \{z_1, z_2, z_3\}$.
11: **end if**
12: $t = t + 1$.
13: **end while**
14: **return** $\arg\min_{x\in P} f(x)/g(x)$

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 (smallest f, largest g, smallest ratio f/g)

Output: the solution with the smallest ratio

Ratio optimization

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

Transformation:

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

⇩

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

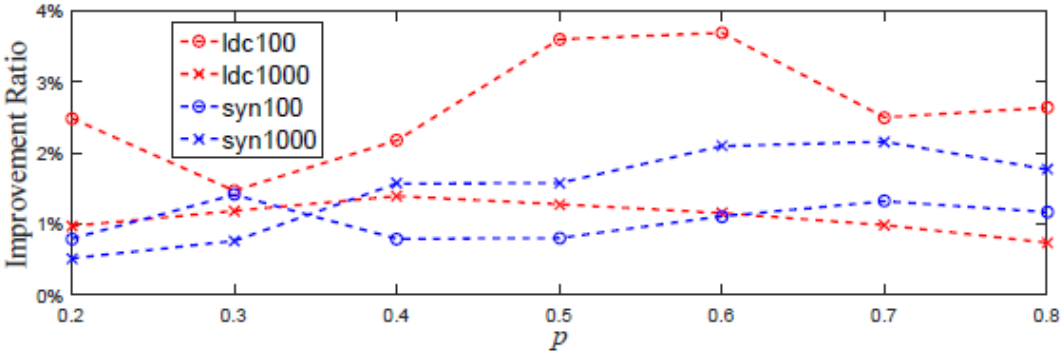
---

Theory: PORM achieves the best-known approximation guarantee $\frac{|X^*|}{(1+(|X^*|-1)(1-\kappa))\gamma}$, previously obtained by GreedRatio [Bai et al., ICML'16]

Application:

F-measure maximization in information retrieval

Always better



Improvement ratio over GreedRatio

Pareto optimization for subset selection

achieve excellent 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

can be reduced to linear time but with performance loss

[Crawford & Kuhnle, 2019]

A sequential algorithm that cannot be readily parallelized

How can Pareto optimization be applied to large-scale subset selection problems?

Outline

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

Pareto optimization for subset selection

Bi-objective transformation:

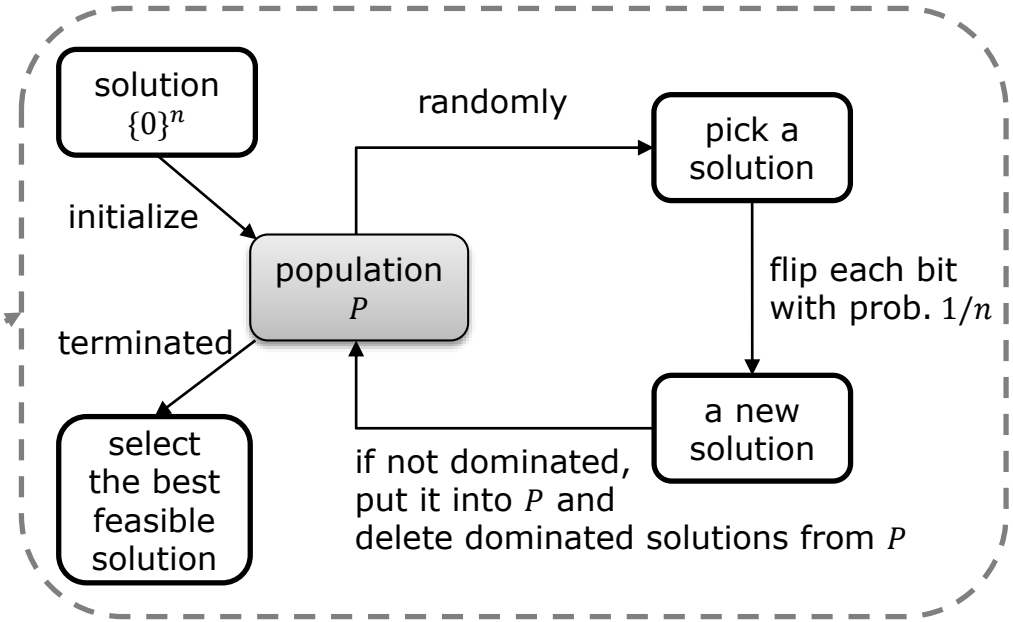
$max_{\boldsymbol{x}\in\{0,1\}^n}\ f(\boldsymbol{x})\quad s.t.\quad |\boldsymbol{x}| \le B$

⇩

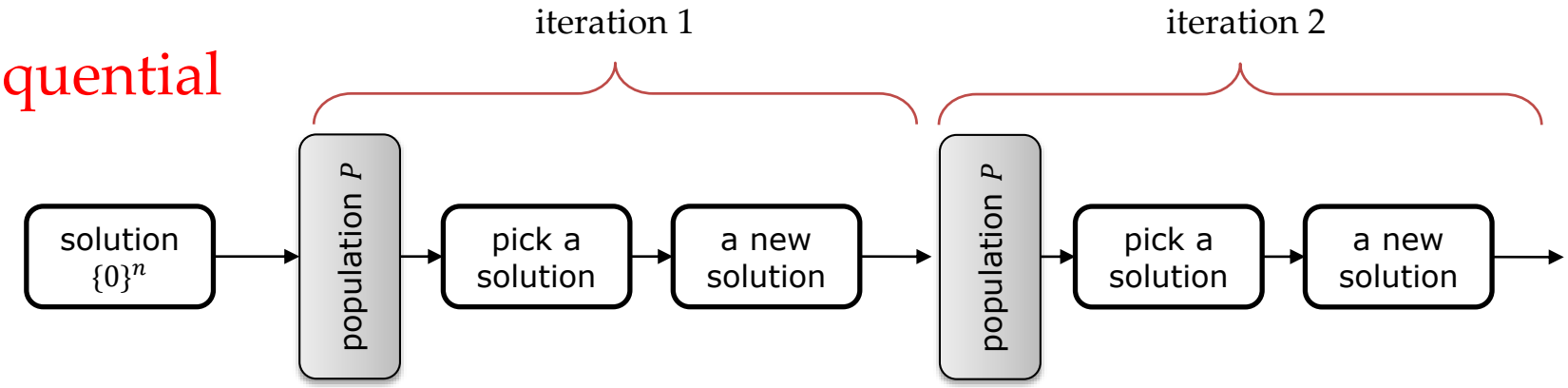
$min_{\boldsymbol{x}\in\{0,1\}^n}\ (-f(\boldsymbol{x}), |\boldsymbol{x}|)$

⇩

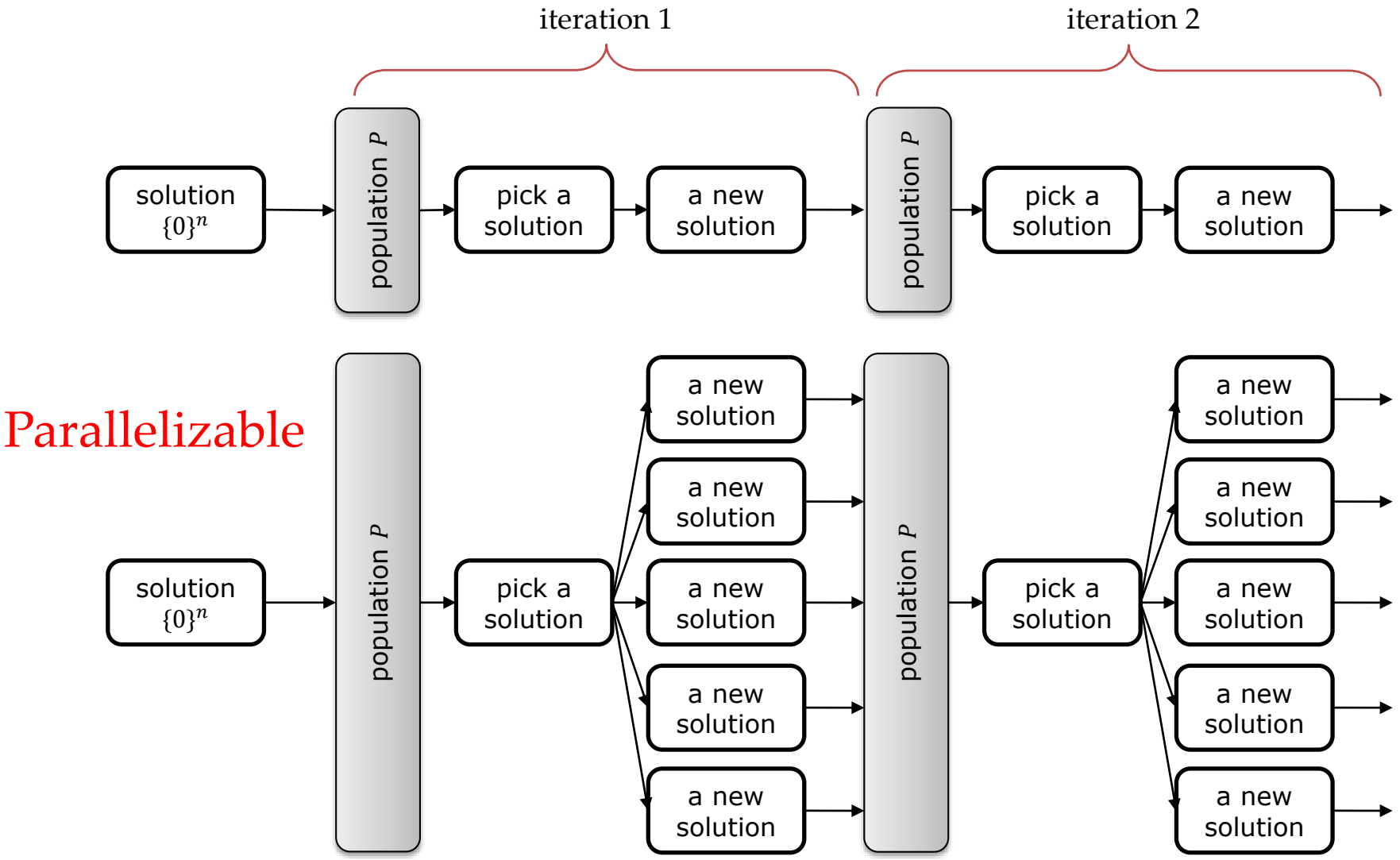
MOEA



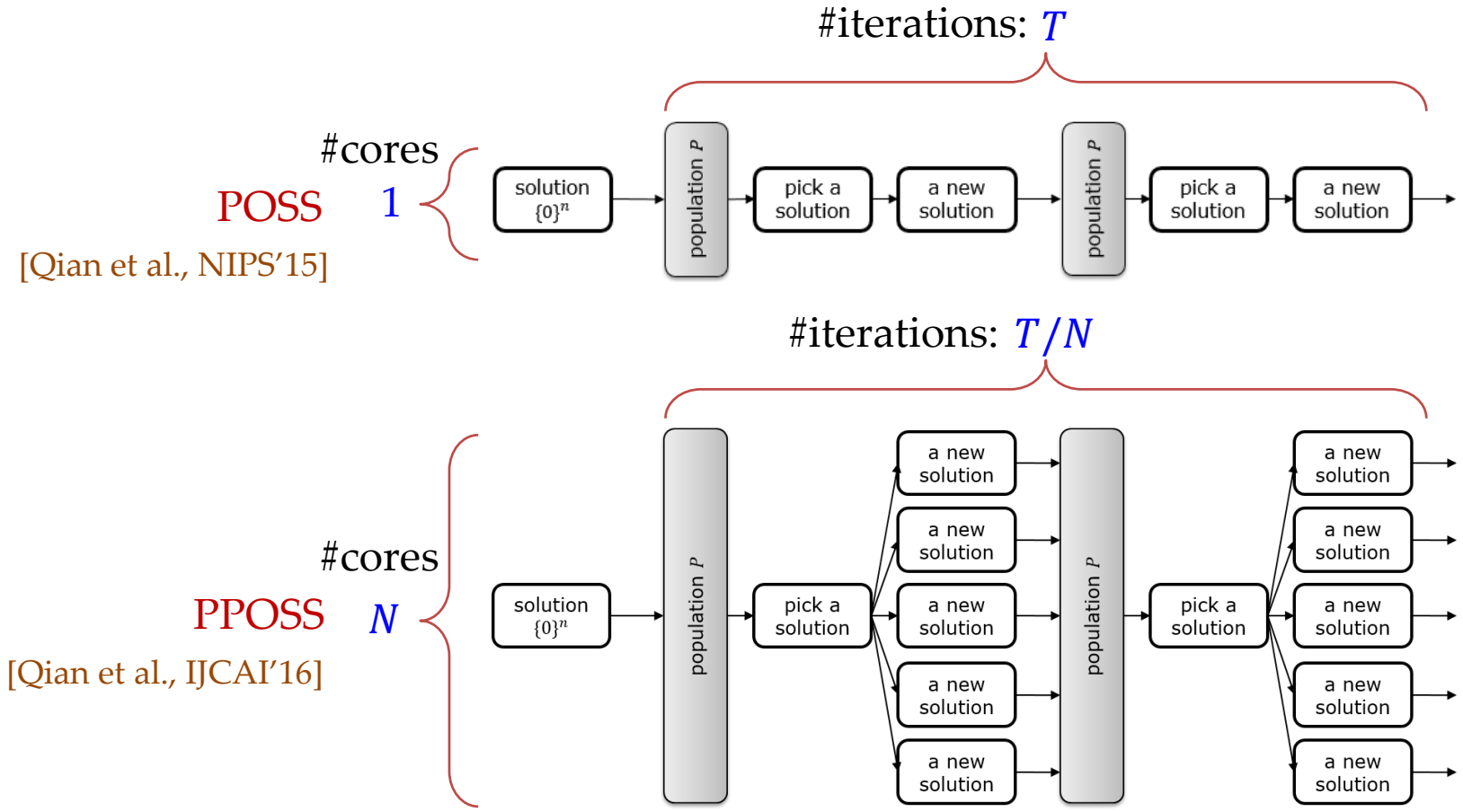
Sequential



Parallel Pareto optimization for subset selection

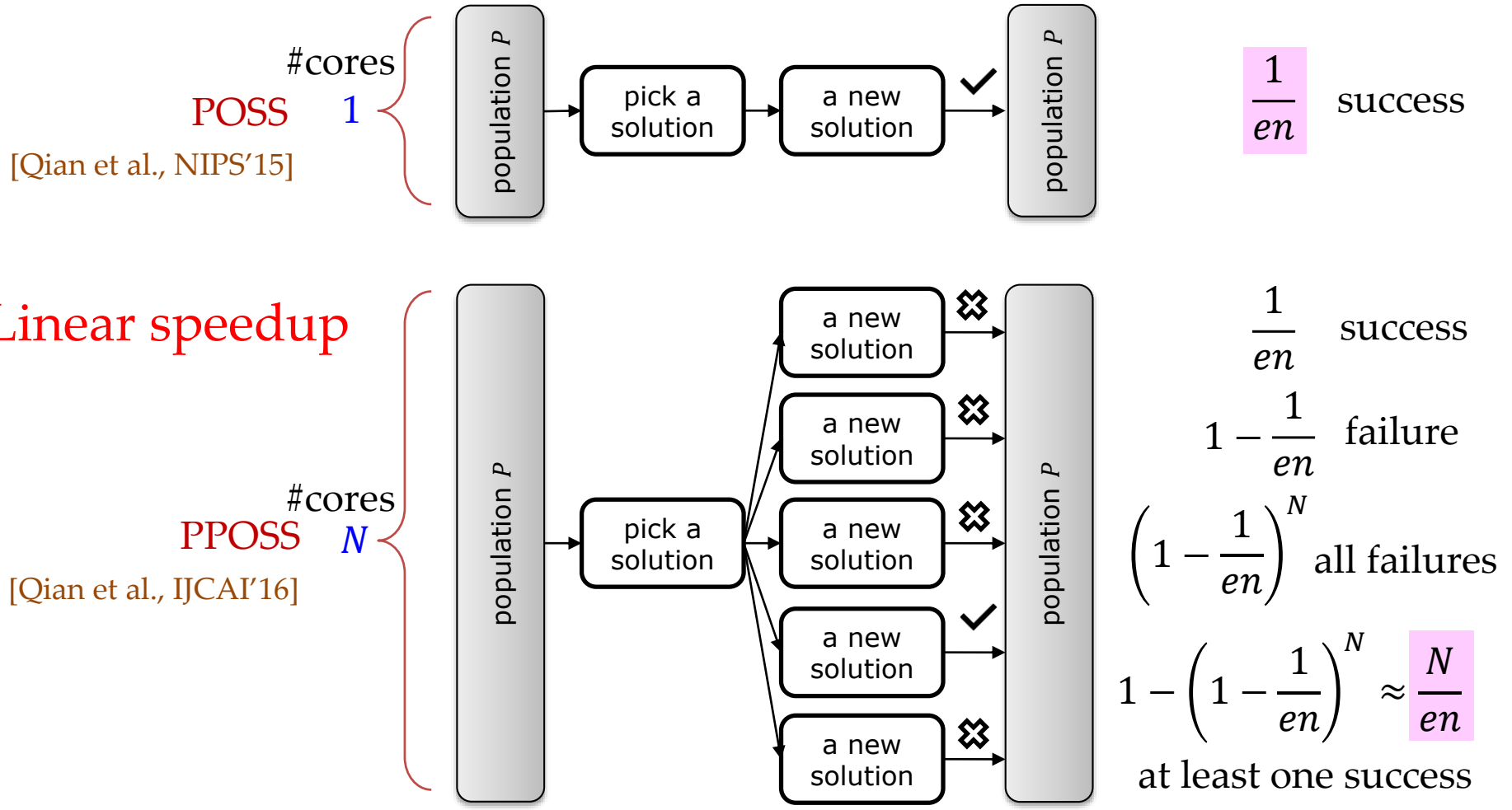


Parallel Pareto optimization for subset selection



Q: the same solution quality? Yes!

Parallel Pareto optimization for subset selection



Theoretical analysis

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

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

(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 $\mathrm{E}[T] = O(1)$.

Keep the optimal approximation guarantee

- When the number N of cores is asymptotically less than the number n of items, the expected number $\mathrm{E}[T]$ of iterations can be reduced linearly w.r.t. the number of cores

Theoretical analysis

Theorem 3. For subset selection with monotone objective function f, the expected number of iterations until PPOSS finds a solution $\boldsymbol{x}$ with $|\boldsymbol{x}| \leq B$ and $f(\boldsymbol{x}) \geq (1 - e^{-\gamma}) \cdot \mathrm{OPT}$ 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 $\mathrm{E}[T] = O(B^2/i)$;

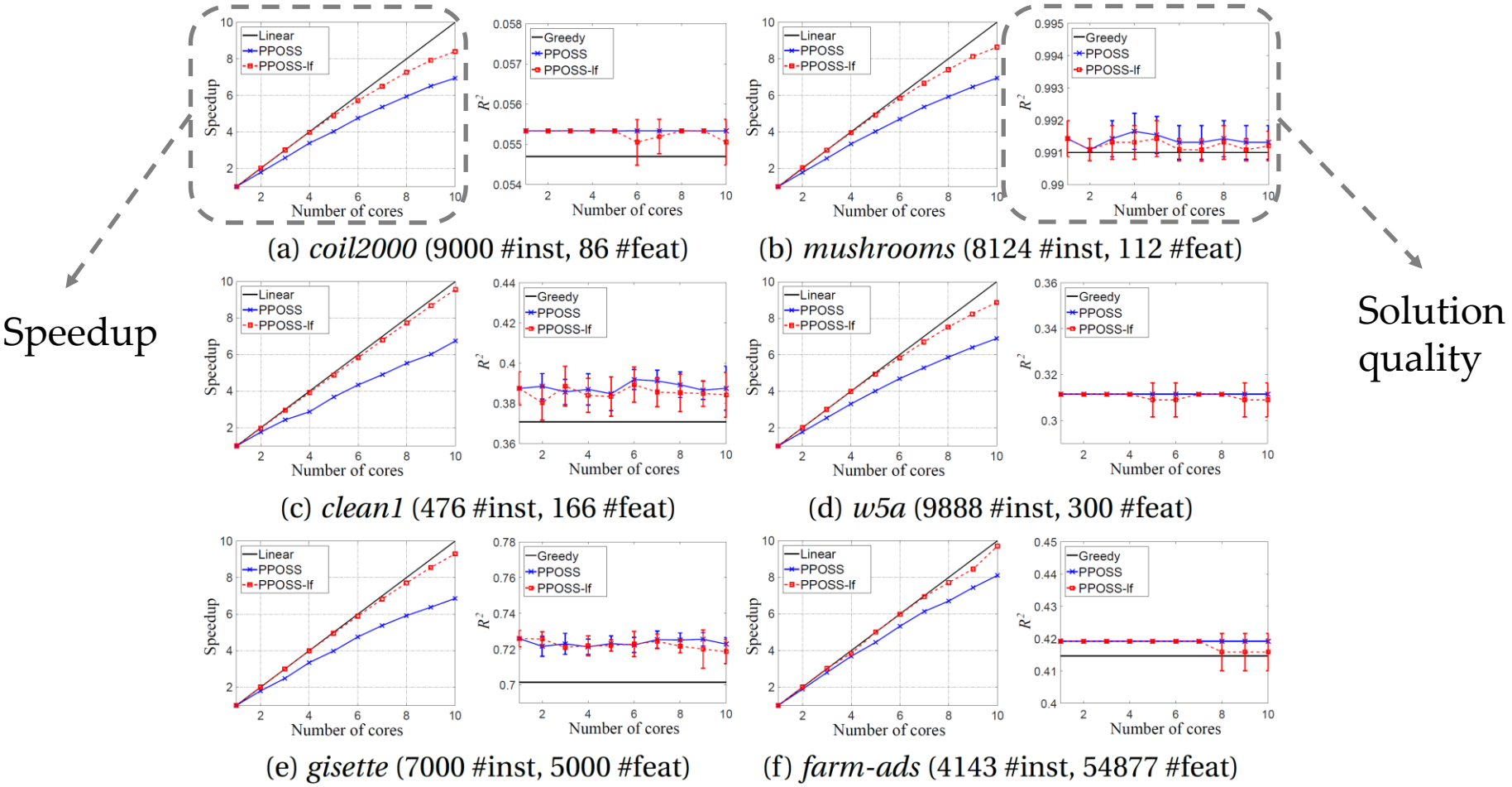
(3) if $N = \Omega(n^{\min\{3B-1,n\}})$, then $\mathrm{E}[T] = O(1)$.

Keep the optimal approximation guarantee

- When the number N of cores is asymptotically less than the number n of items, the expected number $\mathrm{E}[T]$ of iterations can be reduced linearly w.r.t. the number of cores
- With increasing number N of cores, the expected number $\mathrm{E}[T]$ of iterations can be continuously reduced, eventually to a constant

Experiments on sparse regression

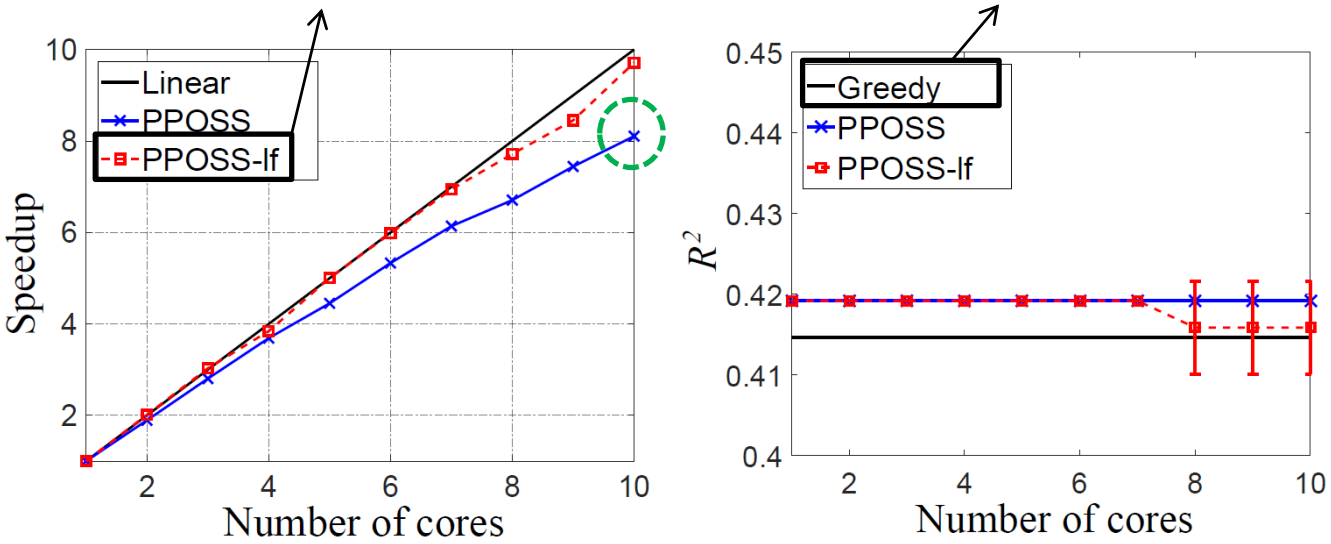
Compare the speedup as well as the solution quality measured by R^2 values with different number of cores



Experiments on sparse regression

the lock-free version of PPOSS

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



(f) *farm-ads* (4143 #inst, 54877 #feat)

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

PPOSS-lf (red line): achieve better speedup as expected; the R^2 values are slightly worse

Pareto optimization for subset selection

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

Parallel Pareto optimization for subset selection

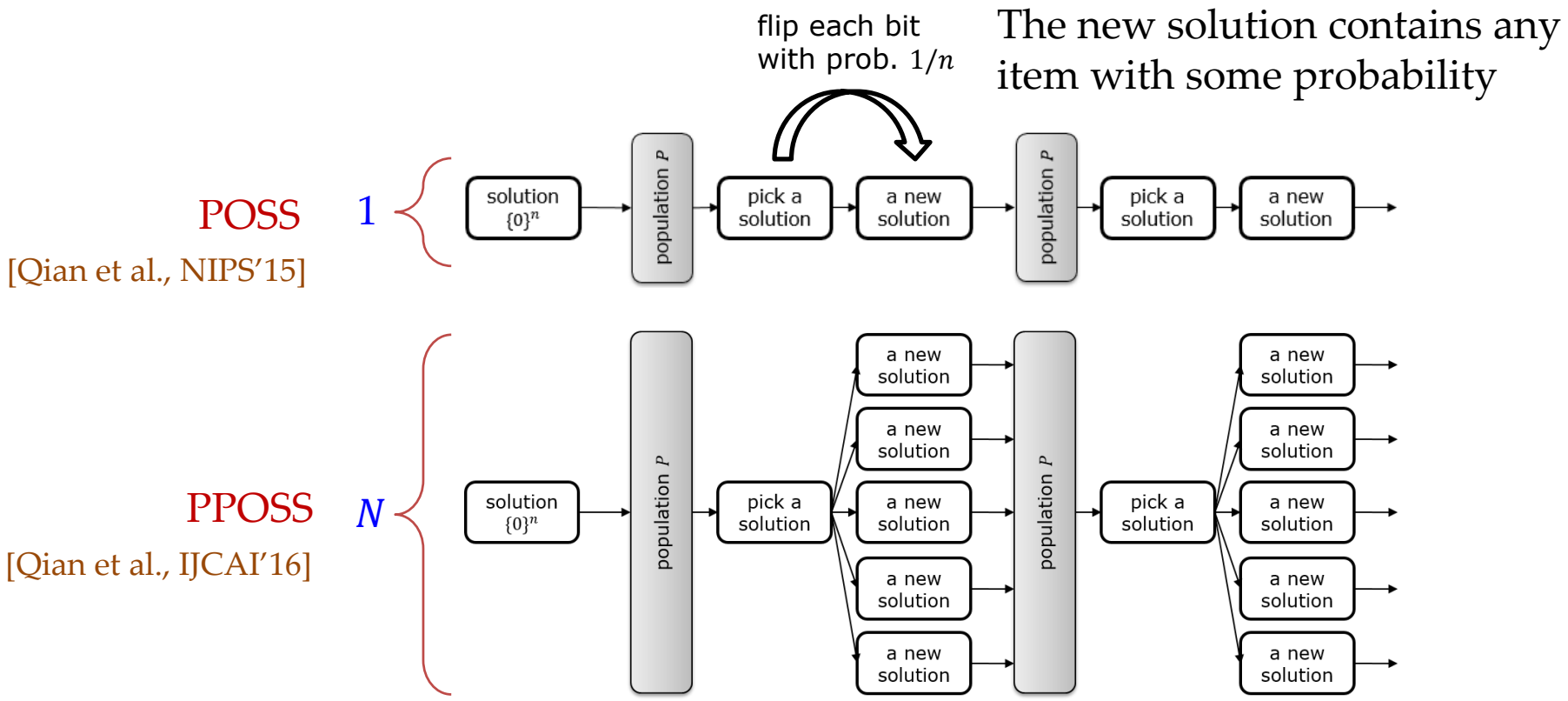
achieve nearly linear runtime speedup while keeping the solution quality

Require centralized access to the whole data set

restrict the application to large-scale real-world problems

Can we make Pareto optimization distributable?

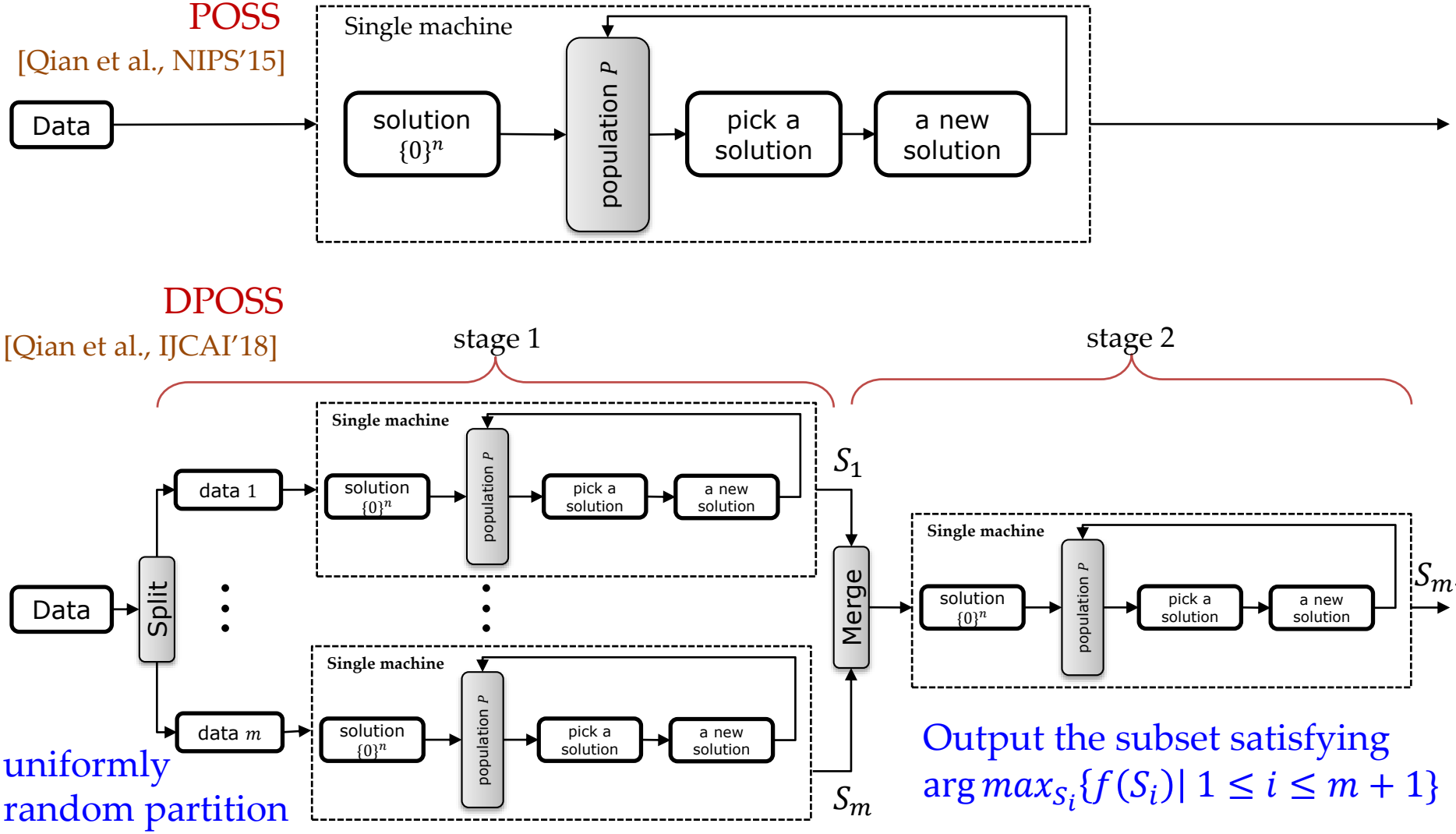
(Parallel) Pareto optimization for subset selection



Require centralized access to the whole data set at each machine

Large-scale data is too large to be stored at one single machine

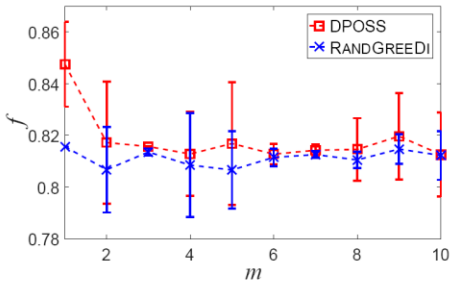
Distributed Pareto optimization for subset selection



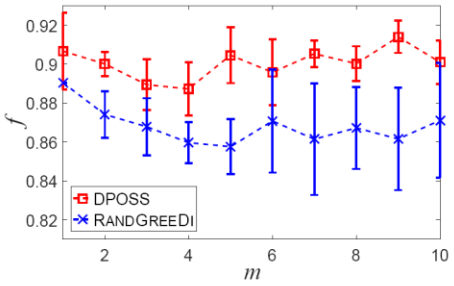
Experiments on sparse regression

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

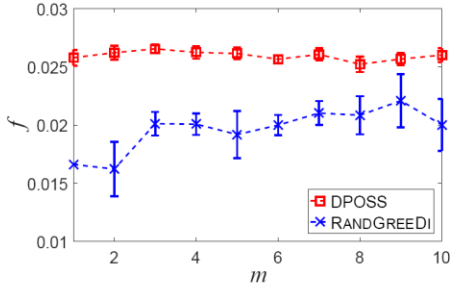
On regular-scale data sets



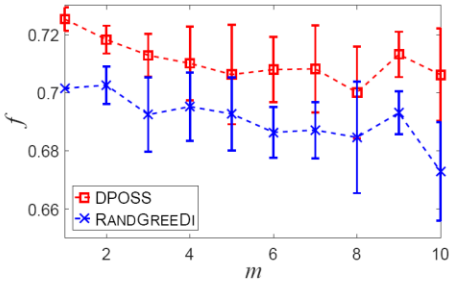
(a) *MicroMass* $(n = 1,300)$



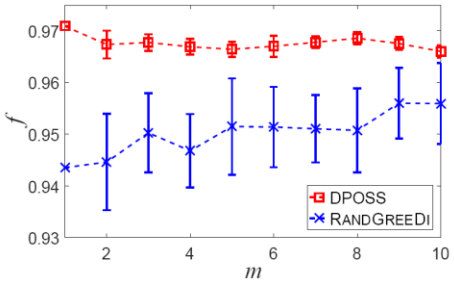
(b) *colon-cancer* $(n = 2,000)$



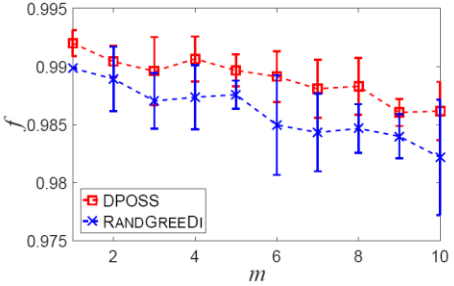
(c) *SVHN* $(n = 3,072)$



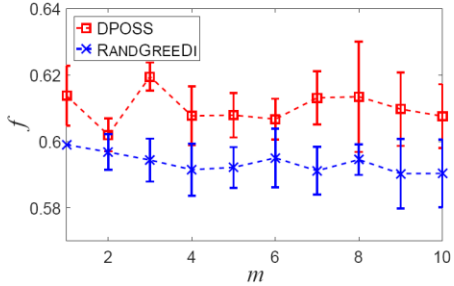
(d) *gisette* $(n = 5,000)$



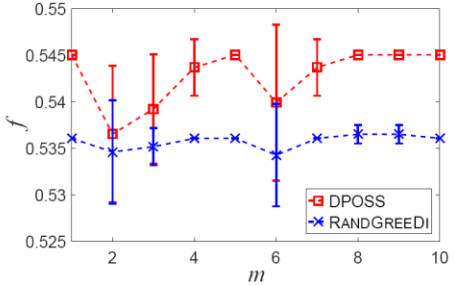
(e) *GHG-Network* $(n = 5,232)$



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



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



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

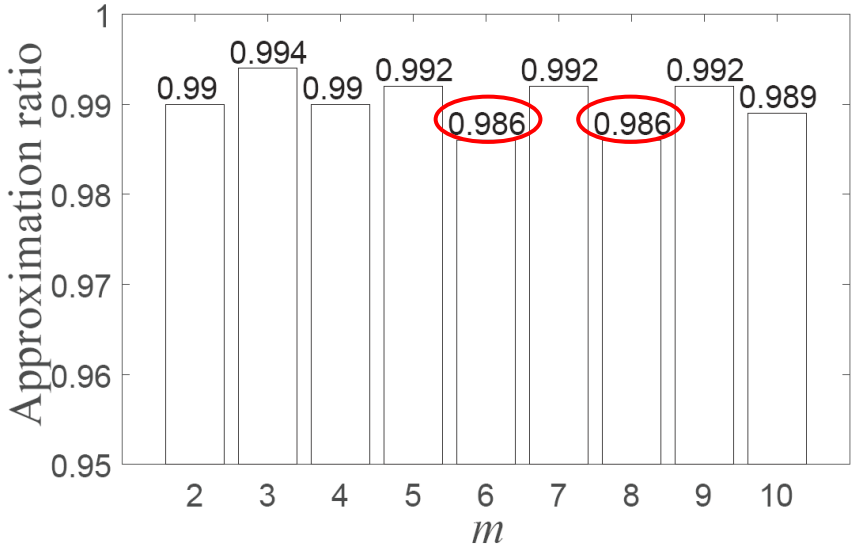
DPOSS is always better than RandGreeDi

Experiments on sparse regression

On regular-scale data sets

$$ratio = \frac{the\ solution\ quality\ by\ \text{DPOSS}}{the\ solution\ quality\ by\ \text{POSS}}$$

DPOSS is very close to the centralized POSS



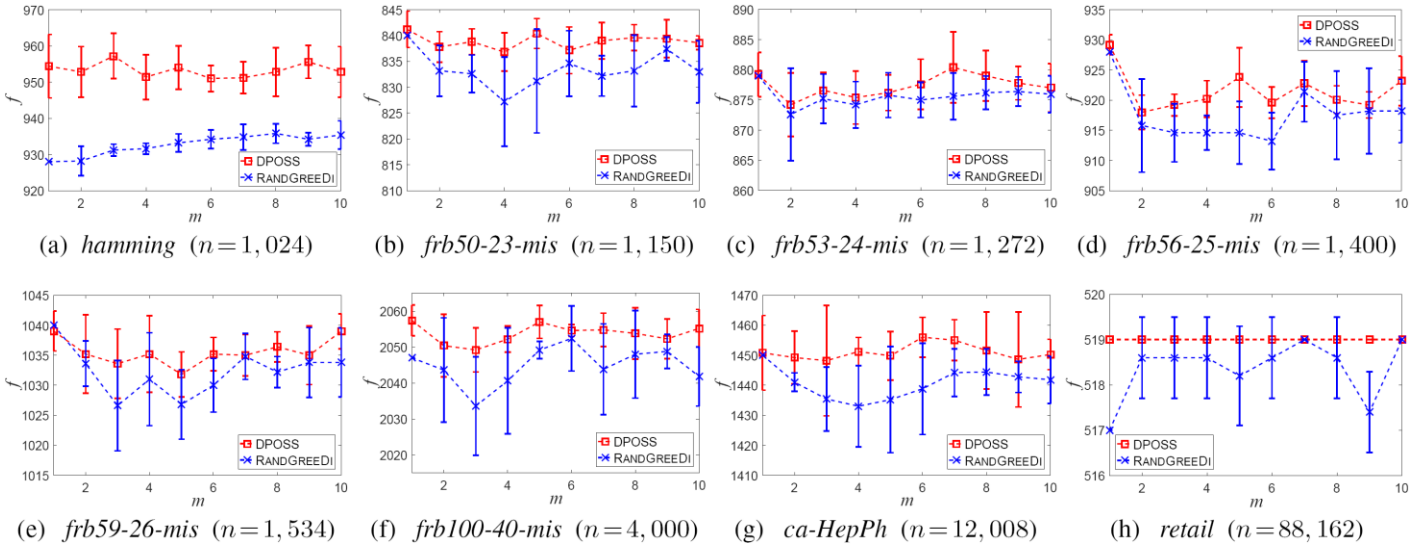
On large-scale data sets

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

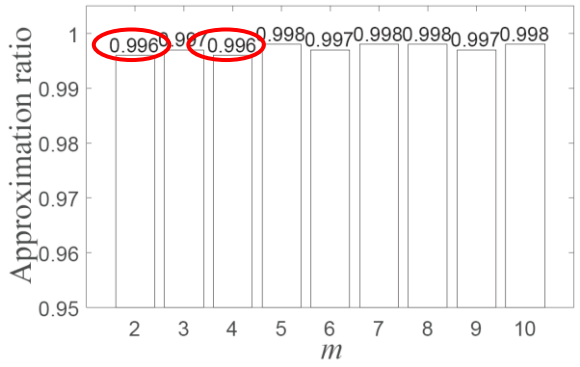
Experiments on maximum coverage

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 excellent 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

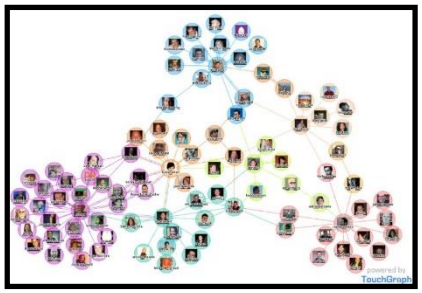
achieve very close performance to the centralized algorithm

large-scale subset selection

Noise

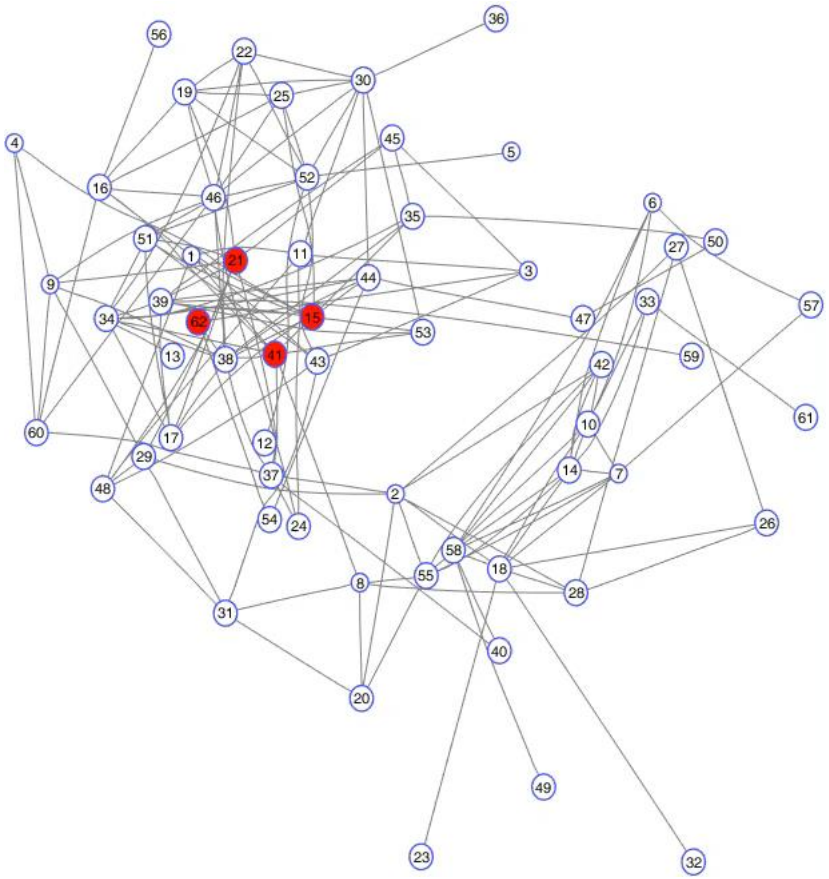
Previous analyses often assume that the exact value of the objective function can be accessed

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 users activated by propagating from X



1st diffusion: 15

2nd diffusion: 16

•
•
•

To achieve an accurate estimation, 10,000 independent diffusion processes are required

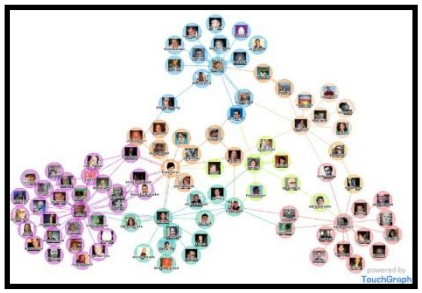
[Kempe et al., KDD'03]

Very expensive

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 users activated by propagating from X

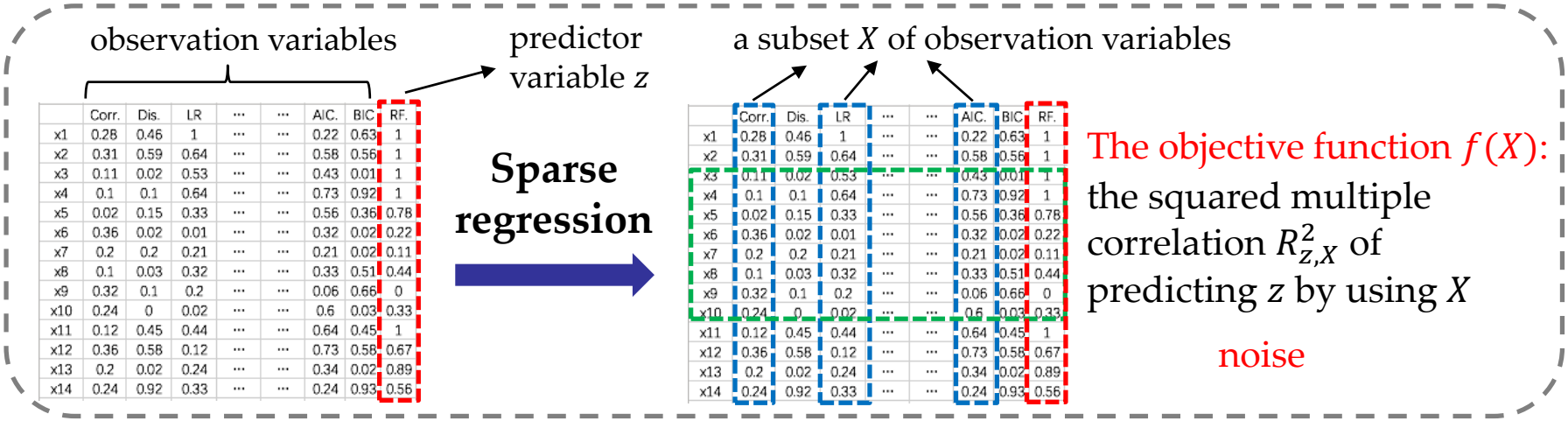
noise

The average number of users activated by a limited number 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



	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

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**
- Pareto optimization for dynamic subset selection
- Conclusion

Noisy subset selection

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

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

exact objective value

noisy objective value

Noise

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

Greedy algorithm & POSS [Qian et al., NIPS'17]:

Multiplicative noise: ϵ: the noise strength

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

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

Additive noise:

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

constant γ

The noiseless approximation guarantee [Das & Kempe, ICML'11; Qian, Yu and Zhou, NIPS'15]

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

a constant approximation ratio

The performance degrades largely in noisy environments

PONSS

In our previous work, threshold selection has been theoretically shown to be robust against noise [Qian et al., ECJ'18]

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

A solution is better if its objective value is larger by least a threshold

Exponentially decrease the running time

POSS [Qian et al., NIPS'15]

"dominate"

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

Reduce the risk of deleting a good solution

PONSS [Qian et al., NIPS'17]

Multiplicative noise:

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

PONSS

In our previous work, threshold selection has been theoretically shown to be robust against noise [Qian et al., ECJ'18]

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

A solution is better if its objective value is larger by least a threshold

Exponentially decrease the running time

"dominate"

POSS [Qian et al., NIPS'15] $X \preccurlyeq Y \Leftrightarrow \begin{cases} f(X) \geq f(Y) \\ |X| \leq |Y| \end{cases}$

Reduce the risk of deleting a good solution

PONSS [Qian et al., NIPS'17]

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

Theoretical analysis

Multiplicative noise:

PONSS ($\theta \geq \epsilon$)

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

better

POSS & Greedy

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

$$\frac{1-\epsilon}{1+\epsilon}\left(1-\left(1-\frac{\gamma}{B}\right)^B\right) = \frac{1-\epsilon}{1+\epsilon}\cdot\frac{\gamma}{B}\cdot\sum_{i=0}^{B-1}\left(1-\frac{\gamma}{B}\right)^i \geq \frac{1-\epsilon}{1+\epsilon}\cdot\frac{\gamma}{B}\cdot\sum_{i=0}^{B-1}\left(\frac{1-\epsilon}{1+\epsilon}\left(1-\frac{\gamma}{B}\right)\right)^i$$

$$= \frac{1-\epsilon}{1+\epsilon}\cdot\frac{\gamma}{B}\cdot\left(1-\left(\frac{1-\epsilon}{1+\epsilon}\right)^B\left(1-\frac{\gamma}{B}\right)^B\right)\Big/\left(1-\frac{1-\epsilon}{1+\epsilon}\left(1-\frac{\gamma}{B}\right)\right)$$

$$= \frac{1}{1+\frac{2\epsilon B}{(1-\epsilon)\gamma}}\left(1-\left(\frac{1-\epsilon}{1+\epsilon}\right)^B\left(1-\frac{\gamma}{B}\right)^B\right)$$

Theoretical analysis

Multiplicative noise:

PONSS ($\theta \geq \epsilon$)

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

Significantly better

POSS & Greedy

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

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

PONSS — a constant approximation ratio

POSS & Greedy — $\Theta(1/B)$ approximation ratio

Theoretical analysis

Multiplicative noise:

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

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

Additive noise:

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

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

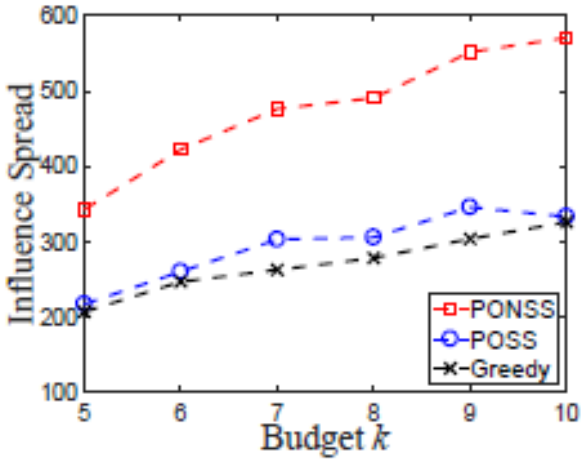
$\frac{2B}{\gamma}-\frac{2B}{\gamma}e^{-\gamma} \geq 2$

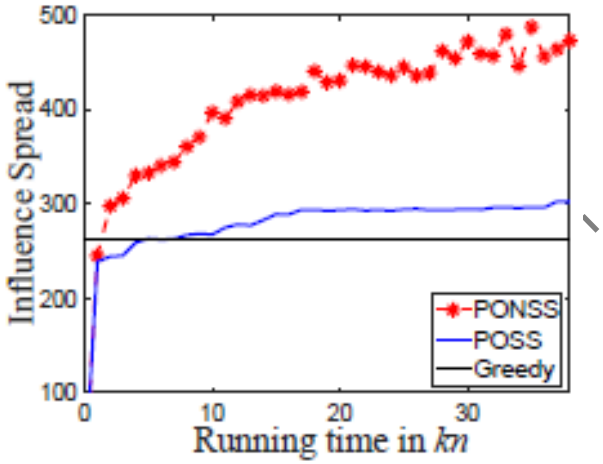
Experimental results - influence maximization

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

Influence spread under different budgets





Performance over runtime

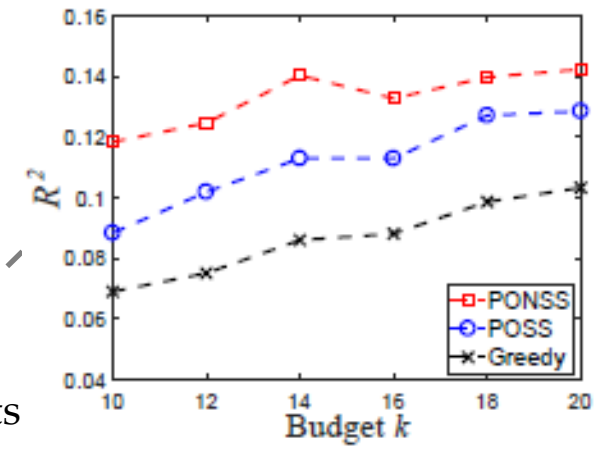
(b) *Weibo* (10,000 #nodes, 162,371 #edges)

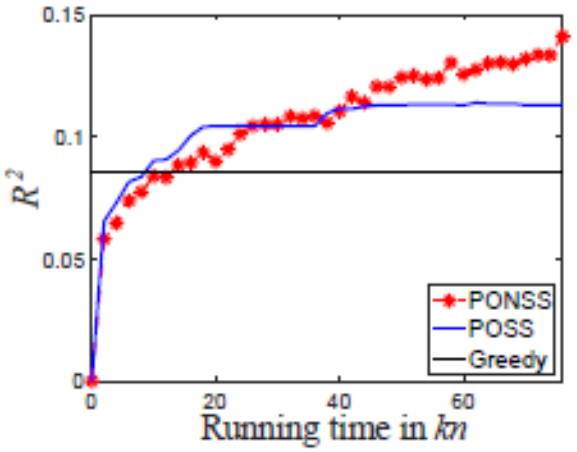
Experimental results - sparse regression

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

R^2 value under different budgets





Performance over runtime

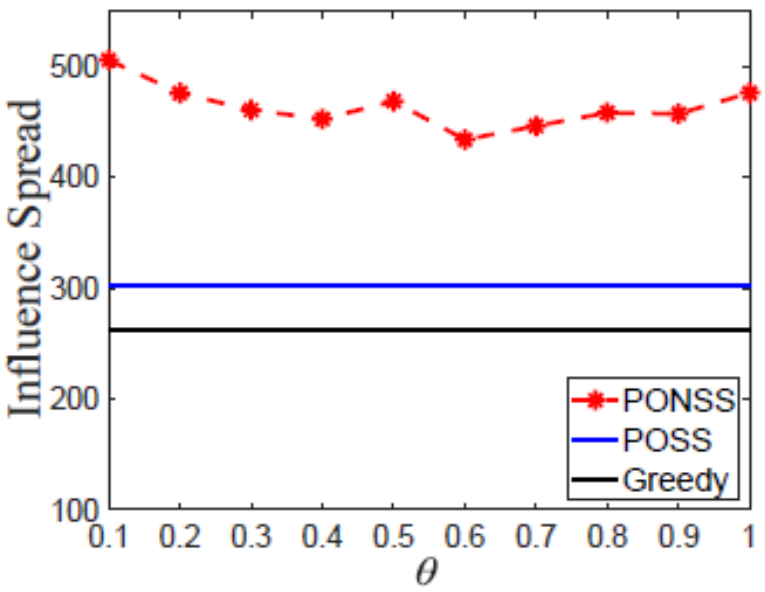
(a) *protein* (24,387 #inst, 357 #feat)

Experimental results – sensitivity to θ

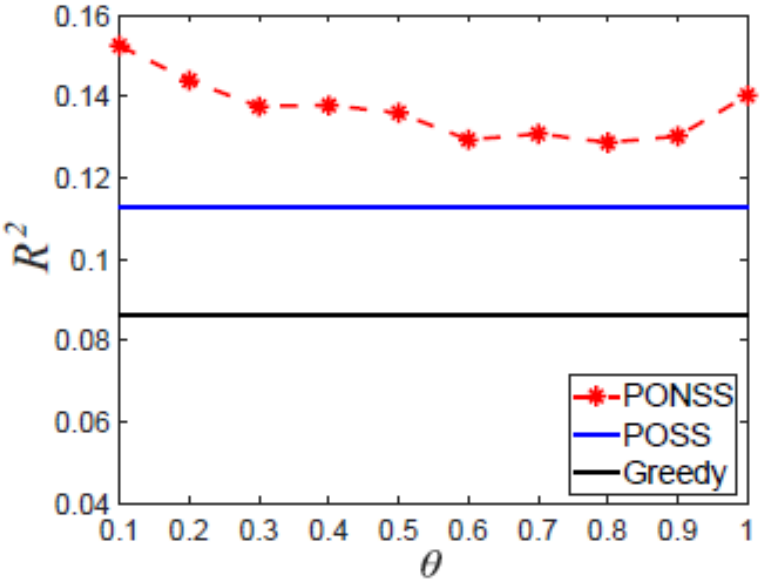
PONSS (red line) vs POSS (blue line) vs Greedy (black line):

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

The performance of PONSS is not very sensitive to θ



(b) *Weibo*



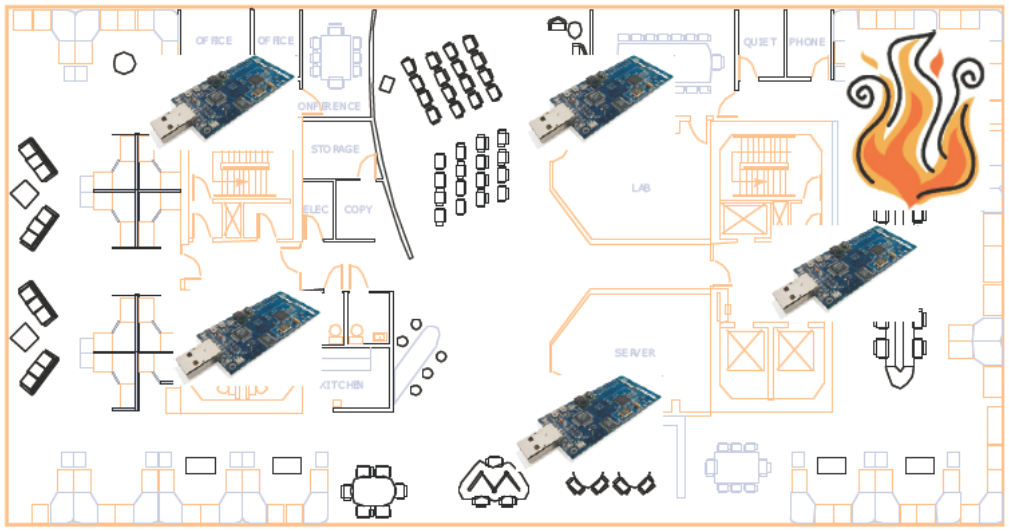
(a) *protein*

Outline

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

Dynamic sensor placement

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



Fire detection

10 sensors

⇩

15 sensors
(more investment)

⇩

12 sensors
(sensor failure)

How about the performance for dynamic subset selection?

Dynamic subset selection

Subset selection with general constraints: given $V = \{v_1, \ldots, v_n\}$, an objective function $f: 2^V \to \mathrm{R}$, a cost function $c: 2^V \to \mathrm{R}$ and a budget B, to find a subset $X \subseteq V$ such that

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

Dynamic subset selection [Roostapour, Neumann, Neumann and Friedrich, AAAI'19]

The available resources may change over time ⇨ The budget B may change over time

To examine: Can an algorithm find a good solution quickly for the new problem, when starting from the solutions obtained for the old problem?

Compare Pareto optimization with the greedy algorithm

Both of them achieve the best-known approximation guarantee for the static problem [Zhang & Vorobeychik, AAAI'16; Qian et al., IJCAI'17]

Greedy algorithm

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

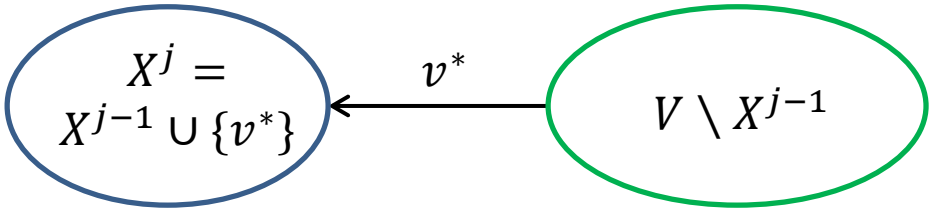
Approximation ratio: $(\alpha/2)(1 - e^{-\alpha})$ [Zhang & Vorobeychik, AAAI'16]

Process: iteratively select one item making the ratio of the increment on f and c maximzied

$$v^* = \arg max_{v \in V \setminus X^{j-1}} \frac{f\left(X^{j-1} \cup \{v\}\right) - f(X^{j-1})}{c(X^{j-1} \cup \{v\}) - c(X^{j-1})}$$

X^j: the subset obtained after j iterations

Iteration j:



Pareto optimization - POMC

Approximation ratio: $(\alpha/2)(1-e^{-\alpha})$ [Qian et al., IJCAI'17]

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

Transformation: ⇩

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

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

Theoretical analysis

[Roostapour, Neumann, Neumann and Friedrich, AAAI'19]

The greedy algorithm can achieve arbitrarily bad approximation ratios during a sequence of dynamic changes

Theorem 1. For dynamic subset selection, there exist instances of dynamically increasing B and decreasing B such that the approximation ratios of the greedy algorithm are $O(1/n)$ and $O(1/\sqrt{n})$, respectively.

POMC can maintain good approximation ratios efficiently

Theorem 2. For dynamic subset selection, with a constant probability, POMC achieves an approximation ratio of $(\alpha/2)(1-e^{-\alpha})$ for any budget $b \in [0, B]$ after $cnP_{max}B/\delta$ iterations. (Already good for decreasing B)

Theorem 3. For dynamic subset selection with B increasing to B^*, with a constant probability, POMC achieves an approximation ratio of $(\alpha/2)(1-e^{-\alpha})$ for any budget $b \in [0, B^*]$ after $cnP_{max}(B^* - B)/\delta$ iterations.

Experimental results - influence maximization

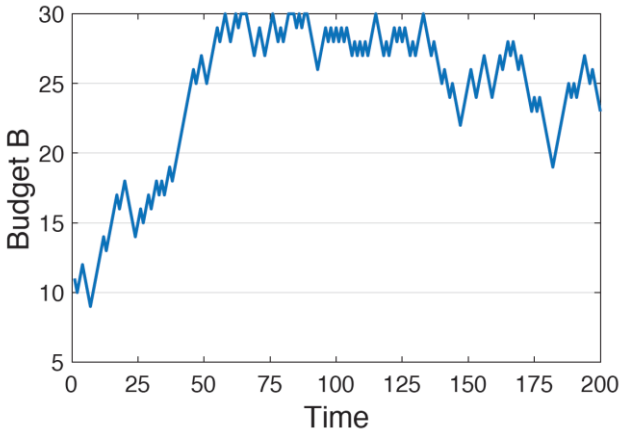
GGA: the greedy algorithm starting from scratch for each new budget

AGGA: the greedy algorithm

POMC$_\tau$: POMC running τ iterations for each new budget

POMC$_\tau^{\mathbf{WP}}$: POMC$_\tau$ with a warm-up phase (running 10,000 iterations for the initial B)

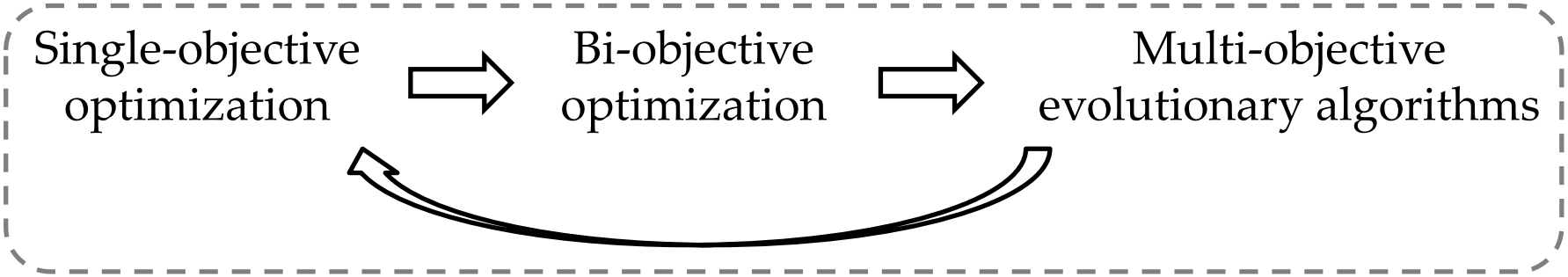
Change of the budget B:



Changes	GGA		AGGA		POMC$_{1000}$		POMC$_{5000}$		POMC$_{10000}$		POMC$^{\mathrm{WP}}_{1000}$		POMC$^{\mathrm{WP}}_{5000}$		POMC$^{\mathrm{WP}}_{10000}$	
	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st	mean	st
1-25	85.0349	12.88	81.5734	14.07	66.3992	17.95	77.8569	18.76	86.1057	17.22	86.3846	10.76	86.9270	12.86	85.8794	14.69
26-50	100.7344	22.16	96.1386	23.99	104.9102	15.50	117.6439	16.71	122.5604	15.54	110.4279	11.08	115.6766	14.21	120.8651	14.97
51-75	118.1568	30.82	110.4893	29.50	141.8249	5.64	155.2126	5.08	158.7228	5.20	140.7838	5.02	149.7658	5.49	157.6169	5.54
76-100	127.3422	31.14	115.2978	27.66	149.0259	3.36	159.9100	3.28	162.7353	3.65	148.3012	3.47	155.1943	4.04	163.1958	3.74
101-125	132.3502	29.62	116.9768	25.45	150.3415	3.17	160.1367	2.81	161.2852	2.68	148.5254	2.67	155.1104	3.05	162.3770	2.81
126-150	134.5256	27.69	118.6962	24.19	147.8998	7.36	154.7319	8.77	154.1470	7.43	143.4908	7.96	150.7567	7.82	156.0363	8.12
151-175	135.7651	25.89	119.4982	22.85	147.2478	4.68	153.1417	5.32	151.2966	3.17	143.2959	4.79	149.5447	4.87	153.2526	3.85
176-200	135.5133	24.41	119.1491	22.04	139.5072	8.08	143.6928	9.16	143.9832	8.67	134.7968	8.72	140.5930	8.61	144.4088	8.08

POMC$_\tau$ achieves better performance than GGA and AGGA after 25 changes, and POMC$_\tau^{\mathrm{WP}}$ can bring improvement in the first 25 changes

Conclusion



- Pareto optimization for subset selection
  - Show excellent performance theoretically and empirically
- Pareto optimization for large-scale subset selection
  - Introduce parallel and distributed strategies
- Pareto optimization for noisy subset selection
  - Introduce noise-aware domination relationship
- Pareto optimization for dynamic subset selection
  - Show robustness against dynamic changes

Future work

- Problem issues
  - Non-monotone objective functions [Qian et al., AIJ'19; Do & Neumann, PPSN'20]
  - Continuous submodular objective functions
  - More complex constraints [Neumann & Neumann, PPSN'20; Do & Neumann, PPSN'20]
  - More uncertain environments
  - Other than subset selection [Neumann & Wegener, NC'06; Friedrich et al., ECJ'10; Neumann et al., Algorithmica'11; Qian et al., IJCAI'15; Crawford, IJCAI'19; Assimi et al., ECAI'20]
- Algorithm issues
  - More complicated MOEAs
- Theory issues
  - Beat the best-known approximation guarantee
- Application issues
  - Attempts on more 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.

For details

- 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.

- 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.

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, 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*, 2018, 22(4), 595-608.

- 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.

For details

- C. Qian, Y. Yu and Z.-H. Zhou. Analyzing evolutionary optimization in noisy environments. *Evolutionary Computation*, 2018, 26(1): 1-41.

- C. Feng, C. Qian and K. Tang. Unsupervised feature selection by Pareto optimization. In: *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19)*, Honolulu, HI, 2019.

- V. Roostapour, A. Neumann, F. Neumann and T. Friedrich. Pareto optimization for subset selection with dynamic cost constraints. In: *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI'19)*, Honolulu, HI, 2019.

- V. Crawford. An efficient evolutionary algorithm for minimum cost submodular cover. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*, Macao, China, 2019.

- V. Crawford and A. Kuhnle. Fast evolutionary algorithms for maximization of cardinality-constrained weakly submodular functions. *arXiv:1908.01230*, 2019.

For details

- C. Qian, Y. Yu, K. Tang, X. Yao and Z.-H. Zhou. Maximizing submodular or monotone approximately submodular functions by multi-objective evolutionary algorithms. *Artificial Intelligence*, 2019, 275: 279-294.

- H. Assimi, O. Harper, Y. Xie, A. Neumann and F. Neumann. Evolutionary bi-objective optimization for the dynamic chance-constrained knapsack problem based on tail bound objectives. In: *Proceedings of 24th European Conference on Artificial Intelligence (ECAI'20)*, Santiago de Compostela, Spain, 2020.

- 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.

- C. Bian, C. Feng, C. Qian and Y. Yu. An efficient evolutionary algorithm for subset selection with general cost constraints. In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*, New York, NY, 2020.

For details

- A. V. Do and F. Neumann. Maximizing submodular or monotone functions under partition matroid constraints by multi-objective evolutionary algorithms. In: *Proceedings of the 16th International Conference on Parallel Problem Solving from Nature (PPSN'20)*, Leiden, The Netherlands, 2020.

- A. Neumann and F. Neumann. Optimising chance-constrained submodular functions using evolutionary multi-objective algorithms. In: *Proceedings of the 16th International Conference on Parallel Problem Solving from Nature (PPSN'20)*, Leiden, The Netherlands, 2020.

- C. Qian. Distributed Pareto optimization for large-scale noisy subset selection. *IEEE Transactions on Evolutionary Computation*, 2020, 24(4): 694-707.

THANK YOU !