Approximation and Linear Programs:

Some approaches

Nguyen Kim Thang
(IBISC, Univ Evry, Paris-Saclay)

JPOC Juin 2019, Metz
Approximation

- **Approximation algorithms**: intractable problems, find the best solution possible (under limited resources)

- **Worst-case paradigm**

  Approximation ratio = \( \max_I \frac{ALG(I)}{OPT(I)} \)
Approximation algorithms: intractable problems, find the best solution possible (under limited resources)

Worst-case paradigm  Approximation ratio = \( \max_I \frac{ALG(I)}{OPT(I)} \)

Mathematical programming: a principled approach
- (Linear) relaxation
- Dual as a lower bound
Approx. ratio vs Integrality gap

\[ \frac{\text{ALG}(I)}{\text{OPT}(I)} \leq \max_I a \text{ lower bound} \]

integrality gap
Given an optimization problem

- **Rounding**
  - construct a linear formulation LP
  - efficiently solve LP and get an optimal fractional solution
  - round the fractional solution to an integer one
Given an optimization problem

- **Primal-Dual**
  - construct a linear formulation
  - construct primal (integer) solution and dual (fractional) solution
  - bound the primal/dual cost

- a lower bound
- dual
- fractional OPT
- OPT
- ALG
Plan

- Iterative Rounding

- Primal-Dual with Configuration LPs
Iterative Rounding
Iterative Rounding: Key lemma

Rank Lemma:

Let $P = \{Ax \geq b, x \geq 0\}$

Assume that $x^*$ be an extreme point solution such that

$$x^*_j > 0 \ \forall 1 \leq j \leq m$$

Then,

the maximal number of linearly independent constraints $A_i x^* = b_i$

equals

the number of variables
**Input:** bipartite graph $G(V_1, V_2)$ with weights on edges

**Output:** a matching of maximum weight

**Formulation**

Formulation 1:

$$x_e = 1 \text{ if the edge is selected}$$

Formulation 2:

$$\begin{align*}
\min & \sum_e w_e x_e \\
\sum_{e \in \delta(v)} x_e & \leq 1 \quad \forall v \\
x_e & \geq 0 \quad \forall e
\end{align*}$$
Lemma:

Assume that $x$ be an extreme point solution such that $x_e > 0 \ \forall e$. Then, there exists $W \subseteq V_1 \cup V_2$ such that:

- $x(\delta(v)) := \sum_{e \in \delta(v)} x_e = 1 \ \forall v \in W$

- the characteristic vectors in $\{\chi(\delta(v)) : v \in W\}$ are linearly independent.

- $|W| = |E|$
Initially, $F \leftarrow \emptyset$

While $E(G) \neq \emptyset$ do

- Find an optimal extreme point solution $x$ of $LP(G)$

- If $x_e = 0$ then update $E(G) \leftarrow E(G) \setminus e$

- If $x_e = 1$ then update $E(G) \leftarrow E(G) \setminus e$, $F \leftarrow F \cup e$
Lemma: there always exists an edge
\[ x_e = 0 \quad \text{or} \quad x_e = 1 \]

Theorem: the matching given by the algorithm is optimal.
Outline of Iterative Rounding

- Formulation of the problem: solvability
- Characterization of optimal (fractional) solution: rank lemma
- Algorithm design: at every step,
  * round some variables to 0 or 1
  * reduce the problem to a sub-problem while maintaining the structure
- Analysis:
  * correctness of the algorithm
  * optimality/approximation
Makespan minimization

**Input**: set of unrelated machines and jobs. Jobs have different processing times on different machines.

**Output**: an assignment job-machine that minimise the maximum load

NP-hard
Given a bound, if there is a feasible assignment with makespan at most the bound

\[ x_{ij} = 1 \] if job \( j \) is assigned to machine \( i \)

\[
\begin{align*}
\min & \quad 1 \\
\text{s.t.} & \quad \sum_i x_{ij} = 1 \quad \forall j \\
& \quad \sum_j p_{ij} x_{ij} \leq T \quad \forall i \\
& \quad x_{ij} \geq 0 \quad i, j
\end{align*}
\]
Lemma:

Assume that $x$ be an extreme point solution s.t. $0 < x_{ij} < 1 \, \forall i, j$.

Then, there exist $J' \subseteq J, M' \subseteq M$ such that:

- $\sum_i x_{ij} = 1 \, \forall j \in J' \quad \sum_j p_{ij} x_{ij} = T \, \forall i \in M'$

- The constraints corresponding to $J'$ and $M'$ are linearly independent

- $|J'| + |M'| = E(G)$
Algorithm

- Initially, $F \leftarrow \emptyset$, $M' \leftarrow M$

- While $J \neq \emptyset$ do

  - Find an optimal extreme point solution $x$ of $LP(G)$. Remove every $(i, j) : x_{ij} = 0$

  - If $x_{ij} = 1$ then update $F \leftarrow F \cup (i, j)$, $J \leftarrow J \setminus j$, $T_i \leftarrow T_i - p_{ij}$

  - If there exists a machine $i$ s.t. $d(i) = 1$

    - or $d(i) = 2$ and $\sum_j x_{ij} \geq 1$

    then $M' \leftarrow M' \setminus i$

- Return $F$
Lemma: the algorithm is well-designed

Theorem: the assignment returned by the algorithm has makespan at most twice the optimum.
Remarks on Iterative Rounding

- Powerful methods: network design, spanning trees, Steiner trees, …

- Recent development:
  
  Nikhil Bansal, On a generalization of iterative and randomized rounding, STOC’19
Primal-Dual with Configuration LPs

[online algorithms, algorithmic game theory N.’19]
Primal-Dual Methods

Principle: dual guides construction of primal solutions.

Designing an algorithm without directly solving

Game: algorithm vs adversary

Unified, simple yet powerful methods
LP-based methods

Given an optimization problem

- **Primal-Dual**
  - construct a mathematical (linear) formulation
  - construct primal (integer) solution and dual (fractional) solution
  - bound the primal/dual cost

Diagram:

- ALG
- OPT
- fractional OPT
- OPT
- a lower bound
- dual

Arrow pointing right from the lower bound.
Survival Routing

**Network**: graph with costs on edges \( c_e : \mathbb{N} \rightarrow \mathbb{R}^+ \)

**Requests**: each request demands \( k \)-edge disjoint paths

**Output**: routing (satisfying the requests) of minimum cost

\[
\sum \limits_e c_e(n_e)
\]
Economies vs Diseconomies

(cost, quantities)

- Economies of scale (sub-modular, etc)
- Diseconomies of scale (convex, etc)

Arbitrarily-grown cost functions
Integrality gap

Natural linear formulation: one request

\[
\begin{align*}
\min \sum_{e=1}^{m} x_{e}^{\alpha} \\
\sum_{e=1}^{m} x_{e} &= 1 \\
x_{e} &\in \{0, 1\}
\end{align*}
\]

\[ OPT = 1 \]

\[ OPT_f = m \cdot \frac{1}{m^\alpha} \]
Configuration LPs: a new way

- Systematically reduce integrality gap for (non-linear) problems.

- Design primal-dual algorithms
  - No need of separation oracles and rounding (typical approaches for configuration LPs)
  - Light-weight algorithms.
A configuration $A$ is subset of requests

\[ x_{ij} = 1 \text{ if request } i \text{ selects strategy } s_{ij} \in S_i \]

\[ z_{eA} = 1 \text{ iff for every request } i \in A, \ x_{ij} = 1 \text{ for some strategy } s_{ij} : e \in s_{ij} \]

\[
\min \sum_{e,A} f_e(A)z_{e,A} \\
\sum_{j : s_{ij} \in S_i} x_{ij} = 1 \quad \forall i \\
\sum_{A : i \in A} z_{eA} = \sum_{j : e \in s_{ij}} x_{ij} \quad \forall i, e \\
\sum_{A} z_{eA} = 1 \quad \forall e \\
x_{ij}, z_{eA} \in \{0, 1\} \quad \forall i, j, e, A
\]
Primal-Dual

\[ \alpha_i = \frac{1}{\lambda} \text{(increase of the total cost due to the request)} \]

\[ \beta_{i,e} = \frac{1}{\lambda} \text{(increase of the cost on the resource if the request uses this resource)} \]

\[ \max \sum_i \alpha_i + \sum_e \gamma_e \]

\[ \alpha_i \leq \sum_{e:e \in s_{ij}} \beta_{ie} \]

\[ \gamma_e + \sum_{i \in A} \beta_{ie} \leq f_e(A) \]
**Primal-Dual**

\[
\min \sum_{e,A} f_e(A) z_{e,A}
\]

\[
\sum_{j:s_{ij} \in S_i} x_{ij} = 1
\]

\[
\sum_{A:i \in A} z_{eA} = \sum_{j:e \in s_{ij}} x_{ij}
\]

\[
\sum_A z_{eA} = 1
\]

\[
x_{ij}, z_{eA} \geq 0
\]

\[
\max \sum_i \alpha_i + \sum_e \gamma_e
\]

\[
\alpha_i \leq \sum_{e:e \in s_{ij}} \beta_{ie}
\]

\[
\gamma_e + \sum_{i \in A} \beta_{ie} \leq f_e(A)
\]

- **Algorithm**: at the arrival of a request, select a strategy that incurs the minimum marginal cost
Smoothness

- **Definition:** A function $f$ is $(\lambda, \mu)$-smooth if

\[
\forall A_1 \subset A_2 \subset \ldots \subset A_n = A, B = \{b_1, \ldots, b_n\}
\]

\[
\sum_{i=1}^{n} [f(A_i \cup b_i) - f(A_i)] \leq \lambda \cdot f(B) + \mu \cdot f(A)
\]

- Similar notion in algorithmic game theory (Roughgarden’15, N.’19)
**Theorem:** Assume that resource cost functions are \((\lambda, \mu)\)-smooth. Then the algorithm is \(\lambda/(1 - \mu)\)-competitive.

**Proof:**

\[
\alpha_i = \frac{1}{\lambda} \quad \text{(increase of the total cost due to the request)}
\]

\[
\beta_{i,e} = \frac{1}{\lambda} \quad \text{(increase of the cost on the resource if the request uses this resource)}
\]

\[
\gamma_e = -\frac{\mu}{\lambda} \quad \text{(the total cost of the resource)}
\]

\[
\max \sum_i \alpha_i + \sum_e \gamma_e
\]

\[
\alpha_i \leq \sum_{e : e \in s_{ij}} \beta_{i,e} \quad \forall i, j
\]

\[
\gamma_e + \sum_{i \in A} \beta_{i,e} \leq f_e(A) \quad \forall e, A
\]
Corollary: If the cost functions are $f(z) = z^\alpha$ then the algorithm is $O(\alpha^\alpha)$-competitive. This is optimal for several problems.

Proof:

The functions is $\left( \Theta(\alpha^{\alpha-1}), \frac{\alpha - 1}{\alpha} \right)$-smooth
Energy-Efficient Scheduling

Energy minimization

**Machine:** unrelated machines, speed scalable

**Jobs:** release $r_j$, deadline $d_j$, volume $p_{ij}$, preemptive non-migration

**Energy:** energy power function is $P(s(t))$, typically $s(t)^\alpha$

**Goal:** complete all jobs and minimize the total energy
Hints

- a strategy of a job is a feasible execution
- a configuration is a feasible schedule
- greedy assignment
Conclusion

- Iterative Rounding

- Primal-dual framework for non-linear/non-convex functions.

- Direction:
  - scheduling with precedence constraints: SDP and non-convex math programming,
  - learning and duality,
  - fairness and duality.

Thank you!