Приближенные схемы

Задачи теории расписаний

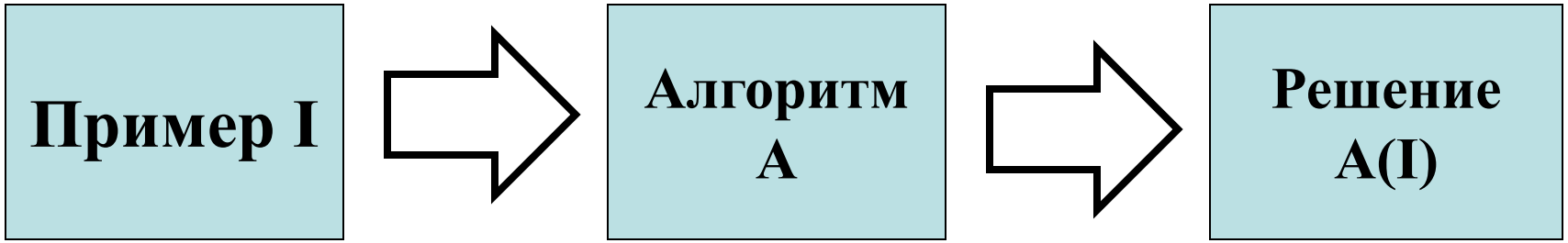
Полиномиальная приближенная схема (PTAS)

Семейство приближенных алгоритмов для задачи П, $\{A_\varepsilon\}_\varepsilon$ называется **полиномиальной приближенной схемой**, если алгоритм A_ε — $(1+\varepsilon)$-приближенный алгоритм и время его работы ограничено полиномом от размера входа при фиксированном ε.

Вполне полиномиальная приближенная схема (FPTAS)

Семейство приближенных алгоритмов для задачи П, $\{A_\varepsilon\}_\varepsilon$ называется **вполне полиномиальной приближенной схемой**, если алгоритм A_ε — $(1+\varepsilon)$-приближенный алгоритм и время его работы ограничено полиномом от размера входа и **$1/\varepsilon$**.

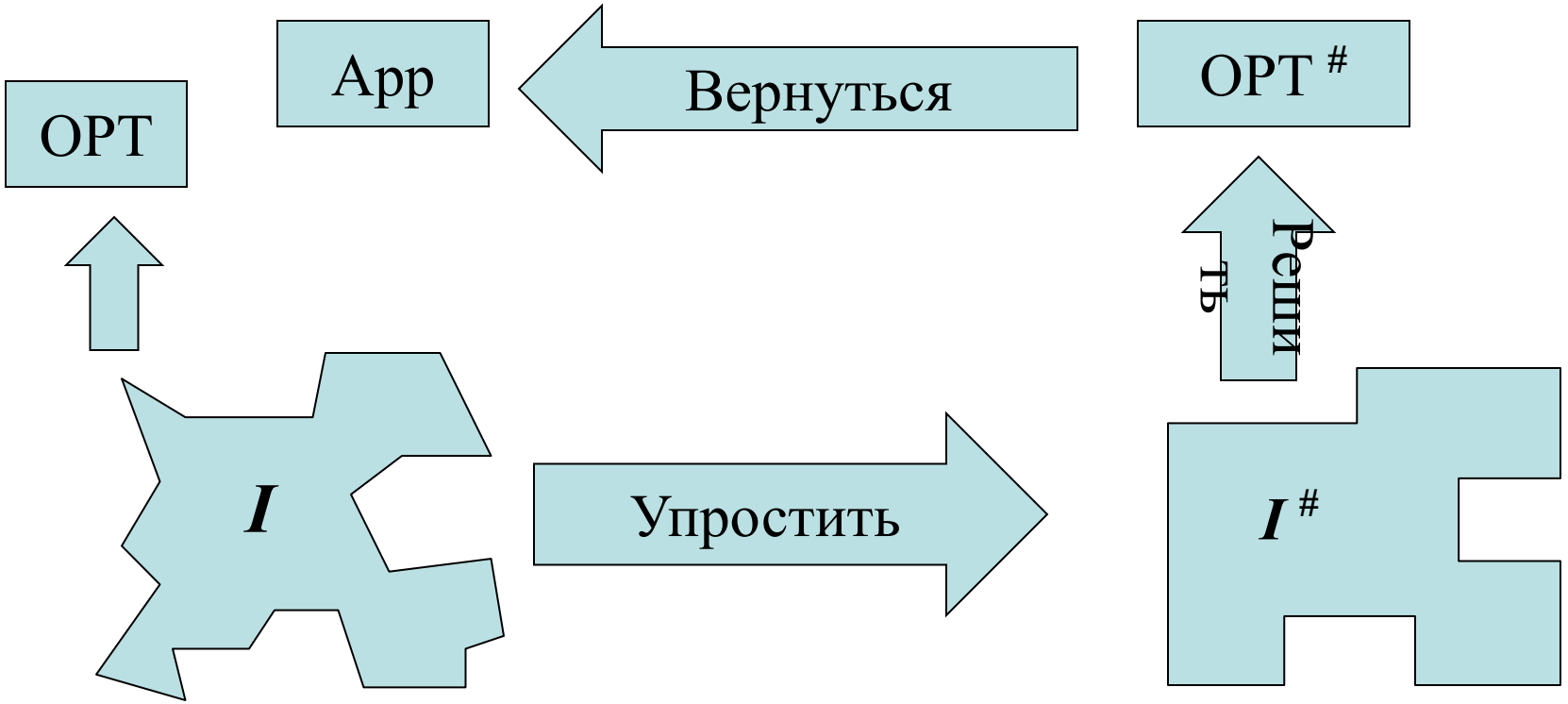
Как построить PTAS



- Упрощение примера ***I***.
- Разбиение пространства **решений**.
- Структурирование работы **алгоритма *A***.

Упрощение примера *I*.

Первая идея превратить трудный пример в более простой в котором легче найти оптимальное решение. Затем мы используем оптимальное решение простого примера для получения приближенного решения для трудного примера.



$P2||C_{max}$

- $J=\{1,..., n\}$ – работы.
- $\{M_1, M_2\}$ – одинаковые машины.
- $j : p_j > 0$ ($i=1,\ldots, n$).
- Каждая работа должна быть выполнена на одной из двух машин.
- Минимизировать длину расписания (C_{max}).
- Прерывания запрещены.
- Каждая машина обслуживает не более одной работы одновременно.

Нижняя оценка

$$p_{sum} = \sum_{j=1}^{n} p_j \qquad p_{\max} = \max_{j=1}^{n} p_j$$

$$C^*_{\max} \geq L = \max\left\{ \frac{p_{sum}}{2}, p_{\max} \right\}$$

Как упростить пример? ($I \rightarrow I^{\#}$)

- **Big** = $\{ j \in J \mid p_j \geq \varepsilon L \}$
  - Новый пример $I^{\#}$ содержит все большие работы из I.
- **Small** = $\{ j \in J \mid p_j < \varepsilon L \}$
  - Пусть $X = \Sigma_{j \in \mathbf{Small}} p_j$.
  - Новый пример $I^{\#}$ содержит $\lfloor X / \varepsilon L \rfloor$ работ длины εL.
- Маленькие работы, как бы склеиваются вместе и разрезаются на маленькие одинаковые кусочки.

Оценка на оптимум

- **Лемма 6.1**

$$\mathrm{OPT}(I^{\#}) \leq (1+\varepsilon)\mathrm{OPT}(I).$$

Доказательство

- X_i – размер всех маленьких работ, выполняемых на машине M_i в оптимальном расписании для I.
- Оставим все большие работы на тех машинах, где они были в оптимальном расписании.
- Заменим все маленькие работы на машине M_i на $\lceil X_i/\varepsilon L \rceil$ работ длины εL.
- $\lceil X_1/\varepsilon L \rceil + \lceil X_2/\varepsilon L \rceil \geq \lfloor X_1/\varepsilon L + X_2/\varepsilon L \rfloor = \lfloor X/\varepsilon L \rfloor$
- $\lceil X_i/\varepsilon L \rceil \varepsilon L - X_i \leq (X_i/\varepsilon L + 1)\, \varepsilon L - X_i \leq \varepsilon L$
- $\mathrm{OPT}(I^{\#}) \leq \mathrm{OPT} + \varepsilon L \leq (1+\varepsilon)\mathrm{OPT}(I)$

Как решить упрощенный пример?

- Как много работ в $I^{\#}$?
- $p_j \geq \varepsilon L$ для всех работ в $I^{\#}$.
- Общая длина всех работ в $I^{\#} \leq p_{sum} \leq 2L$.
- Число работ в $I^{\#} \leq 2L/\varepsilon L = 2/\varepsilon$.
- Число работ в $I^{\#}$ не зависит от n**!**
- Найдем все возможные расписания.
- Число различных расписаний $\leq 2^{2/\varepsilon}$!

Как вернуться к исходной задаче?

- Пусть $\sigma^{\#}$ – оптимальное расписание для $I^{\#}$.
- $L_i^{\#}$ – нагрузка машины M_i в $\sigma^{\#}$.
- $B_i^{\#}$ – общая длина больших работ на M_i в $\sigma^{\#}$.
- $X_i^{\#}$ – размер всех маленьких работ на M_i в $\sigma^{\#}$.
- $L_i^{\#} = B_i^{\#} + X_i^{\#}$.

$$X_1^{\#} + X_2^{\#} = \varepsilon L \left\lfloor \frac{X}{\varepsilon L} \right\rfloor > X - \varepsilon L$$

Процедура ($\sigma^{\#}(I^{\#}) \rightarrow \sigma(I)$)

- Большие работы выполняются на той же машине, что и в $\sigma^{\#}$.
- Выделим интервал длины $X_1^{\#} + 2\varepsilon L$ на машине M_1 и $X_2^{\#}$ на машине M_2.
- Будем последовательно упаковывать маленькие работы в выделенный интервал на M_1, пока не встретим работу, которая туда не поместится.
- Оставшиеся маленькие работы упакуем в выделенный интервал на M_2.

Оценка качества

$$OPT \geq L = \max\left\{\frac{p_{sum}}{2}, p_{\max}\right\}$$

$$L_i^{\#} \leq OPT^{\#} \overset{\text{L 5.1}}{\leq} (1+\varepsilon)OPT$$

$$L_i \leq B_i^{\#} + \left(X_i^{\#} + 2\varepsilon L\right) = L_i^{\#} + 2\varepsilon L \leq$$

$$\leq (1+\varepsilon)OPT + 2\varepsilon OPT = (1+3\varepsilon)OPT$$

Алгоритм PTAS-1

Input ($J=\{1,\ldots, n\}$, $p: J \to \mathbf{Z}^+$)

1) Определим **Big** = $\{ j \in J | p_j \geq \varepsilon L\}$ и
   Small = $\{ j \in J | p_j < \varepsilon L\}$
2) Заменим все маленькие работы на машине M_i
   на $[X/\varepsilon L]$ работ длины εL.
3) Найдем оптимальное расписание $\sigma^{\#}$ в новом примере $I^{\#}$,
   перебрав все допустимые назначения работ по машинам.

- По расписанию $\sigma^{\#}$ построим допустимое расписание σ исходной задачи, используя описанную выше процедуру ($\sigma^{\#}(I^{\#}) \to \sigma(I)$).

Output (σ)

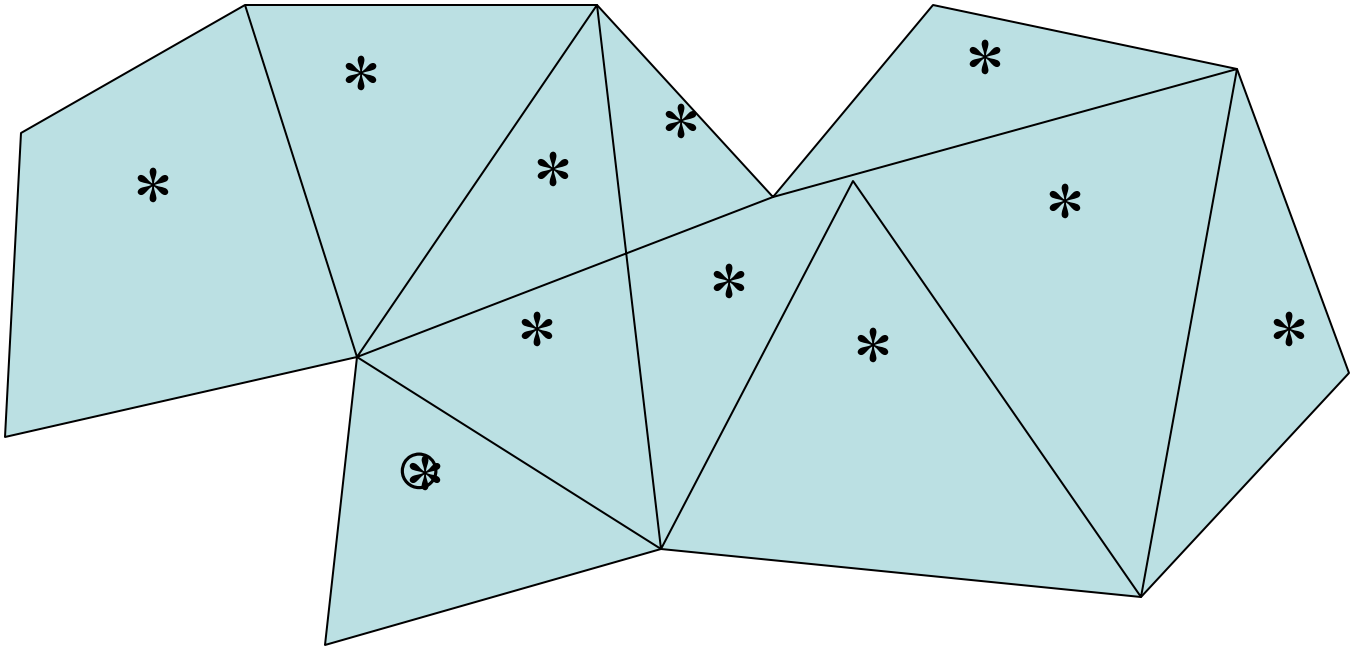
Точность алгоритма PTAS-1

- **Теорема 6.2**

  Алгоритма PTAS-1 – $(1+\varepsilon)$-приближенный алгоритм для задачи $P2||C_{max}$.

Разбиение пространства **решений**

Вторая идея разбить пространство **решений** на несколько меньших областей, в которых проще найти хорошее приближенное решение. Решив задачу отдельно для каждой маленькой области и взяв среди них лучшее есть шанс получить хорошее приближенное решение.



Общая схема

1. Разбиение на области.
2. Выбор «хорошего» представителя в каждой области.
3. Выбор из множества хороших представителей наилучшего.

$P2||C_{max}$

- $J=\{1,..., n\}$ – работы.
- $\{M_1, M_2\}$ – одинаковые машины.
- $j : p_j > 0$ ($i=1,\ldots, n$).
- Каждая работа должна быть выполнена на одной из двух машин.
- Минимизировать длину расписания (C_{max}).
- Прерывания запрещены.
- Каждая машина обслуживает не более одной работы одновременно.

Как разбить на области

- **Big** = $\{ j \in J | \, p_j \geq \varepsilon L \}$
- **Small** = $\{ j \in J | \, p_j < \varepsilon L \}$
- $\boldsymbol{\Phi}$ – множество допустимых расписаний
- Расписание $\sigma \in \boldsymbol{\Phi}$ – назначение работ на машины.
- Определим области $\boldsymbol{\Phi}^{(1)}$, $\boldsymbol{\Phi}^{(2)}$,…согласно назначению больших работ: σ_1 и σ_2 лежат в одной области тогда и только тогда, когда каждая большая работа выполняется в σ_1 на той же машине, что и в σ_2.

Сколько получилось областей?

- Число больших работ $\leq 2L/\varepsilon L = 2/\varepsilon$.
- Число способов разместить большие работы по двум машинам $\leq 2^{2/\varepsilon}$.
- Число различных областей $\leq 2^{2/\varepsilon}$!
- Число различных областей зависит от ε и не зависит от размера входа!

Как выбрать «хорошего» представителя?

- Назначение больших работ зафиксировано в $\boldsymbol{\Phi}^{(l)}$.
- $\mathrm{OPT}^{(l)}$ – длина оптимального расписания в $\boldsymbol{\Phi}^{(l)}$.
- $B_i^{(l)}$ – общая длина больших работ на M_i.
- $T := \max\{B_i^{(1)}, B_i^{(2)}\} \le \mathrm{OPT}^{(l)}$
- Пусть $B_i^{(l)}$ начальная загрузка машины M_i.
- Назначим маленькие работы одну за другой по следующему правилу: каждый раз работа назначается на машину с меньшей нагрузкой на этот момент.
- Полученное расписание $\sigma^{(l)}$ длины $A^{(l)}$ выберем представителем от $\boldsymbol{\Phi}^{(l)}$.

А хорош ли представитель?

1. Если $A^{(I)} = T$, то $A^{(I)} = \mathrm{OPT}^{(I)}$.
2. Пусть $A^{(I)} > T$.
   - Рассмотрим машину с большей загрузкой в $\sigma^{(I)}$.
   - Последняя назначенная на нее работа маленькая и ее длина меньше εL.
   - В момент назначения этой работы загрузка этой машины не превышала $p_{sum} / 2$.
   - $A^{(I)} \leq (p_{sum} / 2) + \varepsilon L \leq (1 + \varepsilon)\mathrm{OPT} \leq (1 + \varepsilon)\mathrm{OPT}^{(I)}$

Алгоритм PTAS-2

Input ($J=\{1,\ldots, n\}$, $p: J \to \mathbf{Z}^+$)

1) Определим **Big** = $\{ j \in J \mid p_j \geq \varepsilon L\}$ и **Small** = $\{ j \in J \mid p_j < \varepsilon L\}$
2) Определим области $\boldsymbol{\Phi}^{(1)}, \boldsymbol{\Phi}^{(2)},\ldots, \boldsymbol{\Phi}^{(h)}$ согласно назначению больших работ по машинам.
3) Сформируем задачи $\boldsymbol{I}^{(1)}, \boldsymbol{I}^{(2)},\ldots, \boldsymbol{I}^{(h)}$, в которых назначение больших работ по машинам фиксировано и задано на входе.
4) Решим каждую из задач $\boldsymbol{I}^{(k)}$, $k = 1,\ldots,h$, назначая маленькие работы одну за другой на машину с меньшей нагрузкой на текущий момент.
5) Пусть σ^* - лучшее из полученных расписаний на шаге 4.

Output (σ^*)

Точность алгоритма PTAS-2

- **Теорема 6.3**

  Алгоритма PTAS-2 – (1+ε)-приближенный алгоритм для задачи $P2||C_{max}$.

Структурирование работы алгоритма

- Основная идея рассмотреть точный, но медленный алгоритм.
- Если алгоритм получает и перерабатывает огромное количество различных значений, то мы можем отбросить часть из них и ускорить работу алгоритма.

$P2||C_{max}$

- $J=\{1,..., n\}$ – работы.
- $\{M_1, M_2\}$ – одинаковые машины.
- $j : p_j > 0$ ($i=1,\ldots, n$).
- Каждая работа должна быть выполнена на одной из двух машин.
- Минимизировать длину расписания (C_{max}).
- Прерывания запрещены.
- Каждая машина обслуживает не более одной работы одновременно.

Краткая запись решения

- Обозначим через σ_k допустимое расписание k первых работ $\{1,..., k\}$.
- Закодируем расписание σ_k с нагрузками машин L_1 и L_2 двумерным вектором $[L_1, L_2]$.
- Пусть V_k обозначает множество векторов, соответствующих допустимым расписаниям k первых работ $\{1,..., k\}$.

Алгоритм «Динамическое программирование»

Input ($J=\{1,\ldots, n\}$, $p: J \to \mathbf{Z}^+$)

1) Положим $V_0=\{[0,0]\}$, $i=0$.
2) **While** $i \neq n$ **do:**

   для каждого вектора $[x,y] \in V_i$ добавить
   вектора $[x + p_i, y]$ и $[x, y + p_i]$ в V_{i+1};
   $i := i + 1$;
3) Найти $[x^*,y^*] \in V_n$, который минимизирует величину $\max_{[x,y] \in V^n} \{x,y\}$

Output ($[x^*,y^*]$)

Трудоемкость алгоритма

- Координаты всех векторов целые числа от 0 до p_{sum}.
- Число различных векторов в V_i ограничено $(p_{sum})^2$.
- Общее количество векторов, вычисляемых алгоритмом не превосходит $n(p_{sum})^2$.
- Трудоемкость алгоритма $O(n(p_{sum})^2)$.
- Размер входа $|I|$ ограничен $O(\log(p_{sum}))=O(\ln(p_{sum}))$ или $O(n \log p_{\max})$.
- Трудоемкость алгоритма не ограничена полиномом от размера входа!

Как упростить множество векторов?

- Все вектора соответствуют точкам плоскости в квадрате $[0, p_{sum}]\times[0, p_{sum}]$.
- Разделим этот квадрат вертикальными и горизонтальными линиями на клетки (квадратики).
- В обоих направлениях линии имеют координаты Δ^i, где $\Delta = 1+ (\varepsilon/2n)$, $i = 1, 2, \ldots, K$.
- $K = \lceil \log_{\Delta}(p_{sum})\rceil = \lceil \ln(p_{sum})/\ln \Delta\rceil = \lceil((1+2n)/\varepsilon) \ln(p_{sum})\rceil$

Отсев векторов

- Пусть два вектора $[x_1,y_1]$ и $[x_2,y_2]$ попали в одну клетку.
- $x_1/\Delta \le x_2 \le x_1\Delta$ и $y_1/\Delta \le y_2 \le y_1\Delta$.
- Из каждой клетки, имеющей непустое пересечение с V_i, выберем один вектор и добавим его в «урезанное» множество векторов $V_i^{\#}$.

Алгоритм FPTAS

Input ($J=\{1,\ldots, n\}$, $p: J \to \mathbf{Z}^+$)

1) Положим $V_0^{\#}=\{[0,0]\}$, $i=0$.
2) **While** $i \neq n$ **do:**

   для каждого вектора $[x,y] \in V_i^{\#}$ добавить

   вектора $[x + p_i, y]$ и $[x, y + p_i]$ в V_{i+1};

   $i := i + 1$;

   Преобразовать V_i в $V_i^{\#}$.
3) Найти $[x^*,y^*] \in V_n^{\#}$, который минимизирует величину $\max_{[x,y] \in Vn^{\#}} \{x,y\}$

Output ($[x^*,y^*]$)

Трудоемкость алгоритма FPTAS

- Множество векторов в $V_i^{\#}$ содержит не более одного вектора из каждой клетки.
- Всего клеток K^2.
- Трудоемкость алгоритма FPTAS $O(nK^2)$.
- $nK^2 = n\lceil((1+2n)/\varepsilon)\ln(p_{sum})\rceil^2$
- Алгоритм FPTAS – полиномиален от размера входа и $1/\varepsilon$.

Оценка на вектора в V_i и $V_i^{\#}$

- **Лемма 6.4**

Для каждого вектора $[x,y] \in V_i$ существует вектор $[x^{\#},y^{\#}] \in V_i^{\#}$, такой что $x^{\#} \leq \Delta^i x$ и $y^{\#} \leq \Delta^i y$.

Доказательство (по индукции)

- $i = 1$: $(x_1/\Delta \le x_2 \le x_1\Delta$ и $y_1/\Delta \le y_2 \le y_1\Delta)$
- $i-1 \to i$: Пусть $[x,y] \in V_i$.
- Тогда $\exists\ [a,b] \in V_{i\text{-}1}$, и либо $[x,y] = [a+p_k,b]$, либо $[x,y] = [a,b+p_k]$.
- Тогда $\exists\ [a^{\#},b^{\#}] \in V_{i-1}^{\#}$: $a^{\#} \le \Delta^{i-1}a$, $b^{\#} \le \Delta^{i-1}b$.
- Алгоритм FPTAS строит вектор $[a^{\#}+p_k,b^{\#}]$ и выбирает $[\alpha,\beta]$, такой что $\alpha \le \Delta(a^{\#}+p_k)$ и $\beta \le \Delta b^{\#}$.
- Имеем $\alpha \le \Delta(a^{\#}+p_k) \le \Delta^i a + \Delta p_k \le \Delta^i(a+p_k) = \Delta^i x$ и $\beta \le \Delta^i y$.

Точность алгоритма FPTAS

- **Теорема 6.5**

  Алгоритма FPTAS – (1+ε)-приближенный алгоритм для задачи $P2||C_{max}$.

Доказательство

$$\max\left\{x^{\#}, y^{\#}\right\} \overset{L\,5.4}{\leq} \max\left\{\Delta^n x, \Delta^n y\right\} = \Delta^n \max\left\{x, y\right\} = \Delta^n OPT$$

$$\left(1+\frac{z}{n}\right)^n \leq 1+2z \Leftarrow \left(0 \leq z \leq 1\right)$$

$$\Delta^n OPT = \left(1+\frac{\varepsilon}{2n}\right)^n OPT \leq \left(1+\varepsilon\right) OPT$$

Практика

- При построении PTAS-1 маленькие работы склеивались в одну и разрезались на одинаковые кусочки. Рассмотрим другой способ построения упрощенного примера. Рассмотрим множество маленьких работ. Если в нем есть две работы длины меньше чем εL склеим их, то есть две работы с длительностями p', $p'' < \varepsilon L$ заменим одной работой с длительностью $p' + p''$. Продолжим склеивание до тех пор пока в примере останется не больше одной работы с $p < \varepsilon L$. Приведет ли такое упрощение к приближенной схеме?
  - Выполняется ли аналог леммы 6.1 ?
  - Можно ли оценить число работ в упрощенном примере ?
  - Как трансформировать оптимальное решение упрощенного примера в приближенное решение исходного примера?