Отчёт по задаче Рюкзак

Сапожников Денис

Я написал удобный класс Knapsack со множество методов:

- solve_recover_dp(W) — решает рюкзак методом ДП и восстанавливает ответ техникой divide-and-conqueror. $O(nW)$ времени, $O(n)$ памяти.
- solve_recover_meet_in_the_middle(W) — решает рюкзак методом meet-in-the-middle. $O(n2^{\frac{n}{2}})$ времени и памяти.
- solve_recover_branch_bound(W) — решает рюкзак методом Branch&Bound. $O(2^n)$ времени в худшем случае. Верхняя граница считается $\frac{1}{2}$-аппроксимацией. Порядок предметов – по стоимости. Порядок ветвей – если взять предмет + $\frac{1}{2}$-аппроксимация больше, чем не взять + аппроксимация, то сначала идём в ветвь, где берём предмет, иначе – сначала в ветвь, где не берём.
- solve_recover_approximation_half(W) — половинная аппроксимация с лекции.
- solve_recover_approximation_slow(W, ε) — первый медленный алгоритм ε-аппроксимации, работает за $O(n \log n + \frac{n^2}{\varepsilon})$, взят с лекции.
- solve_recover_approximation_ibarra(W, ε) — алгоритм Ибарры ε-аппроксимации, за $O(n \log n + \frac{n}{\varepsilon^2})$, взят с лекции.
- solve_recover_on_center_optimization($W, kL, kR, middle\_solver, *args$).

  Предположительно, оптимальный ответ устроен очень просто, если отсортировать все предметы по удельной стоимости $\frac{c}{w}$: $[\underbrace{1, 1, \ldots, 1}_{\textit{левая часть}}, \underbrace{1, 0, 0, 1, \ldots, 1}_{\textit{центральная часть}}, \underbrace{0, 0, \ldots, 0}_{\textit{правая часть}}]$.

  Пусть m_{half} — граница $\frac{1}{2}$-аппроксимации, тогда выберем границы центральной части как $[m_{half} - kL; m_{half} + kR]$ и отправим этот набор предметов в $middle\_solver(W - w_{left}, *args)$.

Результаты экспериментов:

Потестовые результаты экспериментов доступны в приложенном csv-файле.

- Техника Branch&Bound работала очень долго даже при $n = 30$, поэтому этот эксперимент вышел неудачным. Возможно, стоило хорошо аппроксимировать начальное приближение, чтобы отсечь много веток сразу, но я не успел попробовать.

- Как оказалось, медленный алгоритм аппроксимации давал результат лучше, чем алгоритм Ибарры при одном и том же ε (и даже при больших ε).

- Алгоритм с оптимизацией блока по центру вышел лучше всех, и стал выдавать хорошие результаты даже при meet-in-the-middle с параметрами $kL = 15, kR = 30$.

  Техника Branch&Bound работала долго и в центральной оптимизации.

  Но вот медленная аппроксимация выдала очень хорошие результаты, и итоговое решение выделяет блок по центру с параметрами $kL = 100, kR = 500$ и аппроксимирует его с точностью 0.05%. Кроме того, было замечено, что при меньших границах иногда результат бывает лучше, поэтому запускается ещё и центральная оптимизация с параметрами $kL = 50, kR = 250$ и точностью 0.05%. Ну и для уверенности в себе ещё запускается алгоритм Ибарры с точностью 5% (меньшие значения работали так же по приближению, но дольше, поэтому я решил выдать больше времени на центральную оптимизацию) и центральная оптимизация с meet-in-the-middle($kL = 15, kR = 30$). Среди всех подходов берётся лучший.

id посылки — 49864540.