Федеральное агентство по образованию
Государственное образовательное учреждение высшего
профессионального образования
Омский государственный университет
имени Ф.М.Достоевского

Еремеев А.В.

О связи динамического программирования и многокритериальных эволюционных алгоритмов

Препринт

Омск
2008

А.В.Еремеев О связи динамического программирования и многокритериальных эволюционных алгоритмов: Препринт. - Омск: Омский государственный университет, 2008 - 20 с.

В препринте рассматривается широкий класс задач дискретной оптимизации, разрешимых алгоритмами динамического программирования с матричным представлением. Для данного класса задач предложен многокритериальный эволюционный алгоритм (1+1)-EA и установлена верхняя оценка на среднее число итераций до достижения оптимума.

Аннотация

В препринте рассматривается широкий класс задач дискретной оптимизации, разрешимых алгоритмами динамического программирования с матричным представлением. Для данного класса задач предложен многокритериальный эволюционный алгоритм (1+1)-EA и установлена верхняя оценка на среднее число итераций до достижения оптимума. Полученные общие результаты иллюстрируются на задаче 0/1–рюкзак и задаче коммивояжера.

1 Общая схема динамического программирования

Пусть P – задача NP оптимизации, x обозначает исходные данные индивидуальной задачи, $Sol(x)$ – множество допустимых решений, $f_x : Sol(x) \to R$ – целевая функция. Определение задачи NP оптимизации приводится в приложении. Оптимальное значение целевой функции для индивидуальной задачи x обозначим через $f_x^* = \max_{y \in Sol(x)} f_x(y)$, если P-задача максимизации, либо $f_x^* = \min_{y \in Sol(x)} f_x(y)$, если P–задача минимизации.

Рассмотрим задачу P с точки зрения возможности построения динамического программирования (ДП) для нее. При этом будем пользоваться подходом, аналогичным предложенному Г. Воегингером [8], сохраняя, по возможности, те же обозначения.

Предположим, что для задачи P существует точный алгоритм ДП, работа которого состоит из n стадий (предполагается что n, вообще говоря, может зависеть от исходных данных индивидуальной задачи x. На k-й стадии ДП, $k = 1, \dots, n$ строится множество состояний $\mathcal{S}_k$. Каждый элемент этого множества – вектор вида $S = (s_1, \dots, s_\beta) \in Z^\beta$. Здесь и далее Z обозначает множество целых чисел, дополненных, если нужно, символом ∞. Размерность β может зависеть от индивидуальной задачи x, либо фиксирована при постановке задачи P.

Пусть конечное множество $\mathcal{F}$ отображений вида $F : Z^\beta \to Z^\beta$ зависит от индивидуальной задачи x, то же относится к конечному множеству $\mathcal{H}$ отображений $H : Z^\beta \to R$ (здесь и далее R – множество вещественных чисел). Для каждого отображения $F \in \mathcal{F}$ имеется соответствующее ему отображение $H_F \in \mathcal{H}$.

На этапе инициализации ДП пространство состояний $\mathcal{S}_0$ формируется как конечное подмножество пространства Z^β. На стадии k, $k = 1, \dots, n$ пространство состояний $\mathcal{S}_k$ вычисляется на основе множества $\mathcal{S}_{k-1}$:

$$\mathcal{S}_k = \{F(S): \ F \in \mathcal{F}, \ S \in \mathcal{S}_{k-1}, \ H_F(S) \le 0\}. \tag{1}$$

Как видно из (1), отображения из $\mathcal{H}$ служат для отбрасывания некоторых неперспективных состояний.

Пусть $G : Z^\beta \to R$ – значение целевой функции задачи P, вычисляемое на элементах последнего множества состояний $\mathcal{S}_n$. Ввиду предположения о том, что ДП доставляет точное решение задачи P, имеем $f^*_x = \min\{G(S) : S \in \mathcal{S}_n\}$, если P – задача минимизации, либо $f^*_x = \max\{G(S) : S \in \mathcal{S}_n\}$, если P – задача максимизации.

ДП применяется не только для нахождения оптимального значения целевой функции, но и для вычисления оптимального решения $y^* \in Sol(x)$, $f(y^*) = f^*_x$. Будем предполагать что допустимое решение $Y(S_1, \dots, S_n) \in Sol(x)$ вычисляется на основе заданной последовательности состояний $S_0 \in \mathcal{S}_0, \dots, S_n \in \mathcal{S}_n$, если эта последовательность *допустима*, т.е. $S_k = F_k(S_{k-1})$, $F_k \in \mathcal{F}$, $k = 1, \dots, n$ и $H_{F_k}(S_{k-1}) \le 0$, $k = 1, \dots, n$.

Вычислительная сложность ДП, как правило, сокращается за счет исключения из рассмотрения «неперспективных» состояний, например, с использованием принципа Беллмана [4]. Точность алгоритма при этом сохраняется. В настоящей работе, подобно [8], исключение «неперспективных» состояний осуществляется посредством отношения доминирования на множестве состояний.

Будем говорить, что $EXT_k(S) \in \mathcal{S}_n$ – допустимое продолжение состояния $S \in Z^\beta$, если $EXT_k(S) = F_{n-1}(F_{n-2}(...F_k(S)...))$ для некоторых $F_k, F_{k+1}, \dots, F_{n-1} \in \mathcal{F}$ и для всех $i = k, \dots, n-1$ соответствующее $H_{F_i} \in \mathcal{H}$ дает $H_{F_i}(F_{i-1}(...F_k(S)...)) \le 0$.

Определение 1 *Пусть $S, S' \in Z^\beta$ таковы, что для любого допустимого продолжения $EXT_k(S)$ соответствующее ему продолжение $EXT_k(S')$ доставляет решение, не уступающее по целевой функции первому, т.е., $G(EXT_k(S')) \le G(EXT_k(S))$ в случае задачи минимизации или $G(EXT_k(S')) \ge G(EXT_k(S))$ в случае максимизации.*

В таком случае будем говорить, что на стадии k состояние S доминируется состоянием S' по продолжениям*. Состояние, для которого нет ни одного допустимого продолжения, будем считать доминируемым по продолжениям любым состоянием.*

Если бы проверка отношения доминирования по продолжениям осуществлялась быстро, то для исключения «непреспективных» состояний можно было бы эффективно применять это отношение. Однако, непосредственная проверка доминирования по продолжениям, вообще говоря, требует перебора всех возможных продолжений. Для быстрого исключения «неперспективных» состояний будем пользоваться некоторым просто вычислимым отношением частичного порядка $\preceq_{dom}$. Достаточно предусмотреть, чтобы из соотношения $S \preceq_{dom} S'$ следовало бы доминирование S состоянием S' по продолжениям. Будем говорить, что $S \in Z^\beta$ *доминируется* состоянием $S' \in Z^\beta$, если $S \preceq_{dom} S'$.

Пусть J – некоторое подмножество Z^β. Тогда через $\max\limits_{dom} J$ будем обозначать *множество доминирующих подмножеств* множества J, т.е. для любого $S \in J$ и любого доминирующего множества $\mathcal{T} \in \max\limits_{dom} J$ найдется такое $S' \in \mathcal{T}$, что $S \preceq_{dom} S'$.

Следующее утверждение показывает, что если частичный порядок $\preceq_{dom}$ выбран так, что из доминирования следует доминирование по продолжениям, то в ДП достаточно строить только доминирующие подмножества состояний вместо множеств $\mathcal{S}_k$ для обеспечения точного результата.

Утверждение 1 *Пусть доминирующие множества состояний $\mathcal{T}_k$, $k = 1, \ldots, n$ вычисляются таким образом, что*

$$\mathcal{T}_k \in \max_{dom}\{F(S) :\ F \in \mathcal{F},\ S \in \mathcal{T}_{k-1},\ H_F(S) \le 0\}, \tag{2}$$

и из доминирования следует доминирование по продолжениям. Тогда $\mathcal{T}_n$ содержит состояние, соответствующее оптимальному решению, т.е. $f_x^* = \min\{G(S) : S \in \mathcal{T}_n\}$, если P – задача минимизации, либо $f_x^* = \max\{G(S) : S \in \mathcal{T}_n\}$, если P – задача максимизации.

Доказательство. Предположим противное. Исходя из того, что вычисления по формуле (1) дают состояние, соответствующее оптимальному решению, следует что существуют такое $k \in \{1, \ldots, n\}$ и $S \in \mathcal{S}_k \backslash \mathcal{T}_k$, что некоторое продолжение $EXT_k(S)$ является экстремумом функции

$G(EXT_k(S))$, однако, ни одно из продолжений $EXT_k(S')$, $S' \in \mathcal{T}_k$ не оптимально. Но по определению доминирующего множества, $\mathcal{T}_k$ содержит некоторый элемент S', доминирующий S, а значит, S' также доминирует S по продолжениям и $G(EXT_k(S'))$ также оптимально. Получено противоречие. ∎

Покажем, что следующие три условия на $\preceq_{dom}$ гарантируют, что из доминирования следует доминирование по продолжениям:

C.1. для любых $S, S' \in Z^\beta$, если $S \preceq_{dom} S'$, то $F(S) \preceq_{dom} F(S')$;

C.2. для любых $S, S' \in Z^\beta$, если $S \preceq_{dom} S'$ и $H(S) \leq 0$, то $H(S') \leq 0$;

C.3. для любых $S, S' \in \mathcal{S}_n$, если $S \preceq_{dom} S'$, то в случае минимизационной задачи P, $G(S) \geq G(S')$, а в случае максимизационной задачи P, $G(S) \leq G(S')$.

Утверждение 2 *Пусть выполнены условия C.1-C.3 для частичного порядка $\preceq_{dom}$. Тогда для любых $k \in \{1, \ldots, n\}$ и $S, S' \in Z^\beta$, таких что $S \preceq_{dom} S'$, состояние S' также доминирует S по продолжениям на этапе k.*

Доказательство. Если на этапе k состояние S не имеет допустимых продолжений, то утверждение выполняется тривиально. Предположим, что S имеет допустимое продолжение $EXT_k(S) = F_{n-1}(F_{n-2}(...F_k(S)...))$. Тогда по условию C.1,

$$F_k(S) \preceq_{dom} F_k(S'), \ldots, F_{n-1}(S) \preceq_{dom} F_{n-1}(S'),$$

и так как для каждого $i = k, \ldots, n-1$ соответствующее отображение $H_i \in \mathcal{H}$ дает $H_i(F_{i-1}(...F_k(S)...)) \leq 0$, то по C.2 также и $H_i(F_{i-1}(...F_k(S')...)) \leq 0$. То есть, S' имеет допустимое продолжение $EXT_k(S')$. Наконец, по условию C.3, $G(EXT_k(S'))$ не уступает «по качеству» $G(EXT_k(S))$. ∎

Далее будем считать условия C.1-C.3 выполнеными.

Выражая вычисления по формуле (2) в алгоритмическом виде, приходим к следующей общей схеме ДП:

Алгоритм ДП

1. Инициализировать $\mathcal{T}_0 := \mathcal{S}_0$.
2. Для k от 1 до n выполнять:
3. Положить $\mathcal{T}_k := \emptyset$.
4. Для каждого $S \in \mathcal{T}_{k-1}$ и каждого $F \in \mathcal{F}$ выполнять:
5. Если $H_F(S) \leq 0$ и не существует $S' \in \mathcal{T}_k$, доминирующего $F(S)$, то добавить $F(S)$ в $\mathcal{T}_k$ и удалить состояния в $\mathcal{T}_k$, доминируемые $F(S)$.
6. Конец цикла по S и $\mathcal{F}$.
7. Конец цикла по k.
8. Выбрать $\min\{G(S) : S \in \mathcal{T}_n\}$, либо $\max\{G(S) : S \in \mathcal{T}_n\}$.

Для оценки трудоемкости алгоритмов предположим, что вычисления выполняются в модели RAM и имеют место следующие оценки:

- каждая $F \in \mathcal{F}$ вычислима за время θ_1;
- каждая $H \in \mathcal{H}$ вычислима за время θ_2;
- существование $S' \in \mathcal{T}_k$, доминирующего заданное $S \in Z^\beta$ может быть установлено за время θ_3;
- функция G вычислима за время θ_4;
- начальное множество состояний $\mathcal{S}_0$ вычислимо за время θ_5;
- все множества $S' \in \mathcal{T}_k$, доминируемые заданным $S \in Z^\beta$, могут быть удалены из $\mathcal{T}_k$ за время θ_6;
- мощность множества $\mathcal{F}$ ограничена величиной κ.

Значения $\theta_1, \ldots, \theta_6$ и κ могут зависеть от параметров исходных данных x. Временная сложность алгоритма ДП есть

$$O\left(\kappa(\theta_1 + \theta_2 + \theta_3 + \theta_6) \sum_{k=1}^{n} |\mathcal{T}_k| + \theta_4 |\mathcal{T}_n| + \theta_5\right). \tag{3}$$

Для того, чтобы алгоритм ДП имел полиномиальную трудоемкость относительно длины исходных данных, величины $\theta_1, \ldots, \theta_6$, κ, n и $|\mathcal{T}_k|$ для всех $k = 1, \ldots, n$ должны быть ограничены полиномом от длины исходных данных.

2 Матричное представление ДП

Для простоты далее будем рассматривать отношение $\preceq_{dom}$, где первая координата состояния s_1 играет особую роль (ее мы будем называть *ведущей координатой*): в частичном порядке $\preceq_{dom}$ с ведущей координатой $S \preceq_{dom} S'$ тогда и только тогда, когда $s_1 \preceq_1 s'_1$ и $s_j = s'_j$ для всех $j \neq 1$. Здесь и далее $\preceq_1$ обозначает линейный порядок, определенный для координаты s_1 (как правило это отношение будет обычным нестрогим неравенством $\leq$ или $\geq$).

Предположим, что каждая координата $s_j,\ j > 1$ может принимать значения только из множества $B_j \subset Z,\ j = 2, \ldots, \beta$ в рассматриваемой индивидуальной задаче. Тогда если $\preceq_{dom}$ – частичный порядок с ведущей координатой, то процесс вычислений ДП удобно представлять как заполнение некоторой матрицы M размерности $m \times (n+1)$, где $m = \prod_{j=2}^{\beta} |B_j|$. Строки M соответствуют всевозможным комбинациям координат $s_2, \ldots, s_\beta$ (поэтому будем индексировать строки посредством $r = (s_2, \ldots, s_\beta)$). Столбцы $k = 0, \ldots, n$ соответствуют стадиям ДП.

Пусть далее

$$Q(s_2, \ldots, s_\beta) = \{S' \in Z \times B_2 \times \ldots \times B_\beta : s'_2 = s_2, \ldots, s'_\beta = s_\beta\}$$

обозначает множество состояний, которые могут быть представлены в строке $r = (s_2, \ldots, s_\beta)$ матрицы M.

Элементы M_{rk} заполняются значениями s_1 в процессе вычислений ДП. Если ни одного состояния $S \in \mathcal{T}_k \cap Q(r)$, не построено, то $M_{rk} = NA$, где дополнительный символ NA обозначает отсутствие соответствующих состояний. Будем предполагать, что никакой $s_1 \in Z$ не доминируется символом NA в линейном порядке $\preceq_1$.

С использованием матричной формы ДП приведенный выше алгоритм ДП записывается следующим образом.

Алгоритм ДП в матричной форме

1. Инициализировать $\{M_{r0}\}$ для всех r.
2. Для k от 1 до n выполнять:
3. Положить $M_{rk} := NA$ для всех r.
4. Для всех $r = (s_2, \ldots, s_\beta)$, таких что $M_{r,k-1} \neq NA$ и всех $F \in \mathcal{F}$:
5. Пусть $S = (M_{r,k-1}, r)$, $S' = F(S)$ и $r' = (s'_2, \ldots, s'_\beta)$.
6. Если $H_F(S) \leq 0$ и не $s'_1 \preceq_1 M_{r'k}$, то положить $M_{r'k} := s'_1$.
7. Конец цикла по r.
8. Конец цикла по k.
9. Выбрать $\min\{G(S) : S = (M_{rn}, r),\ M_{rn} \neq NA\}$
   или $\max\{G(S) : S = (M_{rn}, r),\ M_{rn} \neq NA\}$.

Предполагая, что отношение $\preceq_1$ проверяется за константное время, временная сложность приведенного алгоритма оценивается

$$O\left(\kappa(\theta_1 + \theta_2)\sum_{k=1}^{n}|\mathcal{T}_k| + \theta_4|\mathcal{T}_n| + \theta_5\right) = O\left(m(\kappa n(\theta_1 + \theta_2) + \theta_4) + \theta_5\right). \quad (4)$$

Как правило, в процессе работы алгоритма ДП сохраняется дополнительная информация для каждого полученного состояния, чтобы после окончания работы ДП, на стадии «бэктрэкинга», можно было с малыми затратами получить аргумент для отображения Y. Обозначим эти дополнительные данные через Γ_{rk}, и пусть $\Gamma_{rk} = r'$ в том и только том случае, когда состояние $S = (M_{rk}, r) \in \mathcal{T}_k$ было получено из состояния $S = (M_{r',k-1}, r') \in \mathcal{T}_{k-1}$ в ДП посредством некоторой функции $F \in \mathcal{F}$, т.е. $S = F(S'),\ H_F(S') \leq 0$. На стадии «бэктрэкинга» значения Γ используются для восстановления цепочки состояний $S_0 \in \mathcal{T}_0, \ldots, S_n \in \mathcal{T}_n$, начиная с состояния $S_n \in \mathcal{T}_n$, если $\mathcal{T}_n \neq \emptyset$. Пусть $GS_n = \max\{G(S) : S \in \mathcal{T}_n\}$, если P-задача «на максимум» или $G(S_n) = \min\{G(S) : S \in \mathcal{T}_n\}$, если P-задача «на минимум». Тогда, полагая $S_n = (M_{r_n,n}, r_n)$, все предшествующие состояния вычисляем рекуррентно по формулам

$$r_k = \Gamma_{r_{k+1},k+1},\ S_k = (M_{r_k,k}, r_k), \quad k = 1, \ldots, n-1.$$

Допустимое решение $Y(S_1, \ldots, S_n)$, отвечающее такой последовательности $S_1, \ldots, S_n$ обозначим через $Y_{\Gamma M}$.

Заметим, что матрица M^*, получаемая по окончанию вычислений ДП, обладает следующим свойством.

Утверждение 3 *Рассмотрим произвольное* $r \in B_2 \times \ldots \times B_n$ *и* $0 \le k \le n$*. Состояние* $(M^*_{r,k}, r)$ *доминирует любое состояние* $S \in \mathcal{S}_k \cap Q(r)$*, если* $\mathcal{S}_k \cap Q(r) \ne \emptyset$*.*

Доказательство. Пусть $S = (s_1, r) \in \mathcal{S}_k$, т.е. $(s_1, r) = F_k(F_{k-1}(\ldots F_1(M_{r'0}, r') \ldots))$ для некоторого $r', F_1, \ldots, F_k$, и $H_{F_i}(F_{i-1}(\ldots F_1(M_{r'0}, r') \ldots)) \le 0$ для всех $i = 1, \ldots, k$. Требуется показать, что $s_1 \preceq_1 M^*_{rk}$.

Воспользуемся индукцией по k. Для $k = 1$ утверждение вытекает непосредственно из схемы ДП и того факта, что $\preceq_1$ – линейный порядок (на итерации $k = 1$ строится все множество $\mathcal{S}_1$ и нулевой столбец M содержит недоминируемые состояния).

Пусть утверждение верно для $k - 1$. Обозначим $\tilde{S} = F_{k-1}(\ldots F_1(M_{r'0}, r') \ldots) \in \mathcal{S}_{k-1}$ и пусть состояние $S^* = (M^*_{rk}, r)$ строится на итерации k некоторым отображением F^*, т.е. $S^* = F^*(S'), S' \in \mathcal{T}_{k-1}$.

По предположению индукции, состояние $\tilde{S}$ доминируется $(M^*_{\tilde{r},k}, \tilde{r})$, где $\tilde{r} = (\tilde{s}_2, \ldots, \tilde{s}_\beta)$. Тогда, по предположению C.1, $S = F_k(\tilde{S}) \preceq_{dom} F_k(M^*_{\tilde{r},k}, \tilde{r})$ и по условию C.2, $H_{F_k}(M^*_{\tilde{r},k}, \tilde{r}) \le 0$, т.к. $H_{F_k}(\tilde{S}) \le 0$. Согласно вычислительной схеме ДП с матричным представлением, столбец k матрицы M^* будет содержать элемент, представляющий некоторое состояние S^*, доминирующее $F_k(M^*_{\tilde{r},k}, \tilde{r})$, поэтому, $S \preceq_{dom} S^*$. Далее, ввиду того, что $\preceq_{dom}$ является частичным порядком с ведущей координатой, для того, чтобы S и S^* были сравнимыми, оба эти состояния должны принадлежать одному множеству $Q(r)$. Следовательно, $S^* = (M^*_{rk}, r)$, и $S \preceq_{dom} (M^*_{rk}, r)$. ∎

Следствие 1. С учетом того, что $\preceq_1$ – линейный порядок, существует единственная матрица M^* со свойством, установленным в Утверждении 3.

Рассмотрим два примера применения описанного подхода для поиска точного решения.

Задача 0/1-рюкзак. Вход состоит из n пар натуральных чисел (p_k, w_k), $k = 1, \ldots, n$ и натурального числа W (p_k называется полезностью элемента k, а w_k – его весом). Задача состоит в выборе подмножества $K \subseteq \{1, \ldots, n\}$, максимизирующего суммарную полезность $\sum_{k \in K} p_k$, при условии $\sum_{k \in K} w_k \le W$.

Положим $\beta = 2$ и пусть каждое состояние $S = (s_1, s_2) \in \mathcal{S}_k,\ k = 1, \ldots, n$ кодирует частичное решение для первых k индексов, где первая координата отвечает за вес частичного решения, а вторая – за его полезность. Начальное множество $\mathcal{S}_0$ состоит из одного элемента $(0, 0)$.

Множество $\mathcal{F}$ содержит две функции:

$$F_1(s_1, s_2) = (s_1 + w_k, s_2 + p_k),$$

$$F_2(s_1, s_2) = (s_1, s_2),$$

где F_1 соответствует добавлению элемента k к частичному решению, а F_2 соответствует пропуску этого элемента.

Множество $\mathcal{H}$ состоит из двух функционалов

$$H_1(s_1, s_2) = s_1 + w_k - W,$$

$$H_2(s_1, s_2) \equiv 0,$$

соответствующих F_1 и F_2. Определим отношение доминирования $\preceq_{dom}$, полагая $S = (s_1, s_2) \preceq_{dom} S' = (s'_1, s'_2)$ тогда и только тогда, когда $s_1 \geq s'_1$ и $s_2 = s'_2$. Таким образом, задан порядок с ведущей координатой s_1, где $s_1 \preceq_1 s'_1$ тогда и только тогда, когда $s_1 \geq s_2$. Вместо символа NA будем использовать $+\infty$.

Положим $G(S) = s_2$ для любого $S \in \mathcal{S}_n$. Тогда ДП в матричном представлении принимает следующий вид.

Алгоритм ДП для задачи 0/1-рюкзак

1. Инициализировать $M_{00} := 0$, $M_{r0} := +\infty,\ r > 0$.
2. Для всех k от 1 до n выполнять:
3. Положить $M_{rk} := +\infty$ для всех r.
4. Для всех r, таких что $M_{r,k-1} \neq \infty$, выполнять:
5.1. Если $M_{r,k-1} + w_k - W \leq 0$ и $M_{r,k-1} + w_k < M_{r+p_k,k}$, положить $M_{r+p_k,k} := M_{r,k-1} + w_k$.
5.2. Если $M_{r,k-1} < M_{rk}$, положить $M_{r,k} := M_{r,k-1}$.
6. Конец цикла по r.
7. Конец цикла по k.
8. Выбрать $\max\{r : M_{rn} \neq \infty\}$.

Для оценки трудоемкости данного алгоритма заметим, что $\kappa = 2$, и $\theta_1, \theta_2, \theta_4$ – константы. Наконец, для любого $S = (s_1, s_2) \in \mathcal{T}_k$, $k = 1, \ldots, n$

целочисленное значение полезности s_2 не превышает np_{max}, где $p_{max} = \max\{p_k : \ k = 1, \ldots, n\}$. Следовательно, $|B_2| = np_{max}$, и формула (4) дает оценку трудоемкости $O(n^2 p_{max})$.

Другой, по-видимому более известный вариант ДП для рассматриваемой задачи может быть представлен с использованием частичного порядка с ведущей координатой в пространстве состояний, где координата s_1 – полезность частичного решения, а s_2 равна его весу. Множества функций $\mathcal{F}$ и $\mathcal{H}$ определяются аналогично указанным выше, с учетом смены координат. В таком случае, для любого состояния $S = (s_1, s_2) \in \mathcal{T}_k, \ k = 1, \ldots, n$ целочисленное значение веса s_2 не превосходит W, поэтому $|B_2| = W$, и выражение (4) дает известную оценку трудоемкости ДП $O(nW)$.

Задача коммивояжера. Входные данные состоят из полного графа $G = (V, E)$ со множеством вершин $V = \{1, 2, \ldots, N\}$ и неотрицательными длинами ребер $w : E \to R^+$. Требуется найти перестановку множества всех вершин $(v_1, \ldots, v_N)$, такую что длина обхода коммивояжера $\sum_{i=2}^{N} w(v_i, v_{i-1}) + w(v_N, v_1)$ минимальна. Не теряя общности, будем считать что $v_1 = 1$.

В данном случае в схеме ДП полагаем $n = N - 1$, $\beta = N + 1$ и каждое состояние $S = (s_1, \ldots, s_{N+1}) \in \mathcal{S}_k, \ k = 1, \ldots, n$ представляет некоторый путь, состоящий из k вершин, начинающийся в вершине 1 и заканчивающийся в вершине s_{N+1}. При этом ведущая координата s_1 содержит длину представленного пути. Роль координат $s_2, \ldots, s_N$ состоит в индикации вершин которые пройдены в данном пути: $s_i = 1$ тогда и только тогда, когда вершина i принадлежит пути, представленному состоянием S. Для вершины $v_1 = 1$ не требуется индикатора. Таким образом, $\sum_{i=2}^{N} s_i = k - 1$.

$\mathcal{S}_0$ состоит из $N-1$ элемента: для каждого $i = 2, \ldots, N$ имеется состояние $S = (w(1, i), s_2, \ldots, s_N, i)$ где $s_j = 0$ для всех $j \neq i$, и $s_i = 1$.

Множество $\mathcal{F}$ состоит из $N - 1$ функции: $F_l, \ l = 2, \ldots, N$. Для $F_l, \ l = 2, \ldots, N$, полагаем $S' = F_i(S)$ следующим образом:

$$s'_1 = s_1 + w(s_{n+1}, l),$$

$$s'_i = \begin{cases} 1, & \text{if } i = l \\ s_i, & \text{otherwise.} \end{cases}, \quad i = 2, \ldots, n.$$

$$s'_{N+1} = l.$$

Итак, $F_l(S)$ соответствует добавлению вершины l к пути, представленному состоянием S. Множество $\mathcal{H}$ также состоит из $N-1$ функционала

$$H_l(S) = s_l, \quad l = 2, \ldots, N,$$

соответствующих $F_l,\ l = 2, \ldots, N$ и предотвращающих возврат в вершины, уже пройденные однажды в данном частичном решении.

определим отношение доминирования $\preceq_{dom}$, полагая что $S \preceq_{dom} S'$ тогда и только тогда, когда $s_1 \geq s'_1$, и $s_i = s'_i,\ i = 2, \ldots, N+1$. Таким образом, частичный порядок имеет ведущую координату s_1, и можно принять, что $s_1 \preceq_1 s'_1$ в том и только том случае, когда $s_1 \geq s'_1$. Будем использовать символ $+\infty$ вместо NA. Целевая функция $G(S) = s_1 + w(s_{N+1}, 1)$ для любого $S \in \mathcal{S}_n$. Число строк в матрице M равно $m = (N-1)2^{N-1}$.

Получаемый в результате алгоритм ДП воспроизводит известный алгоритм Хельда и Карпа [5]. Его схема имеет следующий вид.

Алгоритм ДП для задачи коммивояжера

1. Инициализировать $M_{r0} := +\infty$.

1.1. Для всех i от 2 до N выполнять:

1.2. $\quad M_{r0} := w(1, i)$ для $r = (s_2, \ldots, s_N, i)$
где $s_j = 0$ при $j \neq i$, $s_i = 1$.

2. Для всех k от 1 до n выполнять:

3. $\quad M_{rk} := +\infty$ для всех r.

4. $\quad$ Для всех r, таких что $M_{r,k-1} \neq \infty$ и для всех $l = 2, \ldots, N$ выполнять:

5.1. $\qquad$ Пусть $r = (s_2, \ldots, s_{N+1})$.

5.2. $\qquad$ Пусть $r' = (s'_1, \ldots, s'_{N+1})$, где $s'_l = 1$, $s'_i = s_i$
для всех $i \neq l$, $s'_{N+1} = l$.

5.3. $\qquad$ Если $s_l = 0$ и $M_{r,k-1} + w(s_{N+1}, l) < M_{r',k}$,
положить $M_{r',k} = M_{r,k-1} + w(s_{N+1}, l)$.

6. $\quad$ Конец цикла по r и l.

7. Конец цикла по k.

8. Выбрать

$$\min\left\{M_{rn} + w(s_{N+1}, 1) : r = (s_2, \ldots, s_{N+1}) \in \{0,1\}^{N-1} \times \{2, \ldots, N-1\}\right\}.$$

Для оценки трудоемкости данного алгоритма заметим, что $\kappa = N-1$, а $\theta_1, \theta_2, \theta_4$ – константы. θ_5 есть $O(m)$ в модели вычислений RAM (состояния и индексы строк для этого должны быть представлены как целые числа).

Таким образом, левая часть (4) дает известную оценку трудоемкости ДП для задачи коммивояжера $O\left(N^2 \sum_{k=1}^{N-1} \binom{N-1}{k} + 2^N\right) = O(N^2 2^N)$.

3 Многокритериальный (1+1)-EA

Пусть имеется задача многокритериальной оптимизации «на максимум» с множеством допустимых решений D и конечным семейством критериев $\mathcal{C}$. Критерии из $\mathcal{C}$ могут принимать значение $-\infty$ и имеют вид $f : D \to R \cup \{-\infty\}$. В качестве эвристического метода для поиска Парето-оптимального решения может быть рассмотрен следующий простейший эволюционный алгоритм, так называемый многокритериальный (1+1)-EA (см., например, [7]).

Алгоритм начинает работу с допустимого решения, построенного случайно или детерменированно с помощью некоторой процедуры инициализации. На каждой итерации в многокритериальном (1+1)-EA по текущему решению X строится новое решение X' с помощью некоторой рандомизированной процедуры мутации: $X' = Mut(X) \in D$. В случае, если по каждому критерию $f,\ f \in \mathcal{C}$ новое решение не уступает прежнему, т.е. $f(X') \geq f(X),\ f \in \mathcal{C}$, то полагаем $X := X'$. В противном случае новое решение отбрасывается и текущее решение остается прежним. Процесс вычислений продолжается до тех пор, пока не сработает некоторый достаточно просто проверяемый критерий остановки (например, ограничение по числу итераций).

Как видно из схемы алгоритма, после достижения Парето-оптимального решения в многокритериальном (1+1)-EA текущее решение X может меняться, но совокупность значений $f(X),\ f \in \mathcal{C}$ остается неизменной.

В частном случае, когда $|\mathcal{C}| = 1$, описанный алгоритм представляет собой известную эвристику *эволюционной стратегии* (1+1)-EA [6], также называемую *алгоритмом локального поиска с пересчетом при неудачном шаге* [2].

4 Сравнение ДП и многокритериального алгоритма (1+1)-EA

Пусть для P существует точный алгоритм ДП с матричным представлением. Без потери общности в настоящем параграфе будем считать, что P

– задача «на максимум» и порядок $\preceq_1$ задается неравенством $\leq$. Вместо символа NA естественно будет использовать $-\infty$.

Перейдем от исходной задачи NP-оптимизации P к задаче отыскания той самой матрицы M^*, которая рассчитывается при работе ДП. Задача отыскания M^* может быть поставлена как задача многокритериальной оптимизации с nm критериями M_{rk}, $r \in B_2 \times \ldots \times B_\beta$, $k = 1, \ldots, n$, в предположении что множество допустимых решений D есть всевозможные способы вычисления элементов M_{rk} с помощью функций $F \in \mathcal{F}$. При этом предполагается, что начальный столбец $M_{r'0}$ – часть исходных данных задачи. Вычисление значений M_{rk} осуществляется последовательно, по возрастанию k.

В данном случае допустимое решение $X = (\Phi, \Gamma) \in D$ состоит из целочисленной матрицы Γ размерности $m \times n$ (см. § 1) и $m \times n$-матрицы Φ, элементы которой суть функции $\Phi_{rk} \in \mathcal{F}$. Предполагается, что для любой пары (r, k) соответствующий элемент Φ_{rk} – та функция, которая должна использоваться, чтобы получить состояние $S = (M_{rk}, r)$ из состояния $S' = (M_{r',k-1}, r')$, где $r' = \Gamma_{r,k}$. В случае, если выбранный для подсчета M_{rk} элемент $M_{r',k-1}$ не определен (т.е. $M_{r',k-1} = -\infty$) или нарушены соотношения

$$\Phi_{rk}(M_{\gamma(r,k),k-1}, \gamma(r,k)) = (M_{rk}, r), \quad H_{\Phi_{rk}}(M_{\gamma(r,k),k-1}, \gamma(r,k)) \leq 0, \qquad (5)$$

где для удобства записи $\Gamma_{r,k}$ обозначено через $\gamma(r, k)$, то полагаем $M_{r,k} = -\infty$.

Таким образом, при заданном начальном столбце $M_{r'0}$ и матрицах Φ и Γ, все элементы $M_{r,k}$, $r' \in B_2 \times \ldots \times B_\beta$ матрицы $M(\Phi, \Gamma)$ вычисляются по формуле

$$M_{rk} = \begin{cases} \Phi_{rk}\left(M_{\gamma(r,k),k-1}, \gamma(r,k)\right)_1, & \text{если } \Phi_{rk}(M_{\gamma(r,k),k-1}, \gamma(r,k)) \in Q(r) \\ & \quad \text{и } H_{\Phi_{rk}}(M_{\gamma(r,k),k-1}, \gamma(r,k)) \leq 0 \\ & \\ -\infty, & \text{иначе,} \end{cases}$$

в цикле по возрастающим значениям $k = 1, \ldots, n$. Указанная процедура вычисления матрицы критериев $M(\Phi, \Gamma)$ имеет трудоемкость $O(mn(\theta_1 + \theta_2))$.

Матрицы $M(\Phi, \Gamma)$ и Γ определяют допустимое решение $Y_{M(\Phi,\Gamma),\Gamma}$ задачи P, если только последний столбец матрицы $M(\Phi, \Gamma)$ не состоит из символов $-\infty$.

Рассмотрим многокритериальный (1+1)-ЕА для данной задачи. Столбец M_{r0} заполняется на стадии инициализации (1+1)-ЕА теми же значениями, что и на шаге 1 алгоритма ДП. Начальное решение строится случайным образом с равномерным распределением всех компонент Φ и Γ.

Будем называть *локальной операцией* следующее изменение решения (Φ, Γ): во-первых, случайным образом с равномерным распределением выбирается пара индексов $r \in B_2 \times \ldots \times B_\beta$ и $k \in \{1, \ldots, n\}$. Во-вторых, значение Φ_{rk} выбирается случайным образом с равномерным распределением из $\mathcal{F}$. В-третьих, значение Γ_{rk} выбирается случайным образом с равномерным распределением из $B_2 \times \ldots \times B_\beta$. Заметим, что число всевозможных исходов локальной операции равно $n\kappa m^2$.

Определим оператор мутации через процедуру локальной операции. Пусть s – целочисленная случайная величина с распределением Пуассона с параметром $\lambda = 1$. Мутация заданного решения состоит в выполнении с ним последовательности из $s + 1$ независимой локальной операции.

Согласно схеме многокритериального (1+1)-EA, на каждой итерации данного алгоритма оператор мутации следует применять к текущему решению (Φ, Γ) с матрицей значений критериев M. Для полученного после мутации решения $(\tilde{\Phi}, \tilde{\Gamma})$ вычисляется новая матрица значений критериев $\tilde{M}$. Если $\tilde{M}_{rk} \geq M_{rk}$ для всех $k = 1, \ldots, n$, $r \in B_2, \ldots, B_\beta$, то текущее решение заменяется на $(\tilde{\Phi}, \tilde{\Gamma})$.

Теорема 1 *Среднее число итераций многокритериального (1+1)-EA до получения матрицы $M = M^*$ оценивается сверху величиной*

$$en^2\kappa m^2(\ln m + 1),$$

где e – основание натурального логарифма.

Доказательство. Рассуждения проводятся аналогично доказательству теоремы 7 в [7]. С учетом единственности матрицы M^* (см. следствие из утверждения 3), после получения $M_{rk} = M^*_{rk}$ для каких-либо r и k, на последующих итерациях (1+1)-EA будет допускаться переход только в такие решения (Φ, Γ), в которых $M_{rk} = M^*_{rk}$. Таким образом, однажды полученные требуемые элементы M_{rk} уже не заменяются иными значениями.

Предположим теперь, что текущее решение (Φ, Γ), обеспечивает $M_{rk} = M^*_{rk}$ для всех $r \in B_2 \times \ldots \times B_\beta$, $k = 1, \ldots, t-1$ для некоторого номера стадии t. Тогда для каждой пары (r, t), такой что $M_{rt} < M^*_{rt}$, имеется элемент $M_{r',t-1} = M^*_{r',t-1}$ и функция $F \in \mathcal{F}$ такие, что $(M^*_{rt}, r) = F(M_{r',t-1}, r')$

и $H_F(M_{r',t-1}, r') \leq 0$ (это следует из схемы алгоритма ДП). В таком случае, мутация, состоящая только в замене Φ_{rt} на F и Γ_{rt} на r' будет непременно принята в (1+1)-EA и ведет к новому текущему решению $(\tilde{\Phi}, \tilde{\Gamma})$, с матрицей значений критериев $\tilde{M}$, где $\tilde{M}_{rt} = M^*_{rt}$. Вероятность такой мутации есть $1/(en\kappa m^2)$, так как ровно одна локальная операция в операторе мутации происходит с вероятностью $1/e$, и требуемая локальная операция выбирается из $n\kappa m^2$ возможных с вероятностью $1/(n\kappa m^2)$. Будем говорить в таком случае, что происходит *успешная* мутация для пары (r, t).

Если в текущем решении имеется q пар (r, t) таких, как описано выше, то вероятность того, что мутация окажется успешной для одной из этих пар будет не менее $q/(en\kappa m^2)$. Число q не превышает m, и каждое из значений $m, m-1, \dots, 1$ величина q для стадии t примет не более одного раза. Следовательно, суммарное число итераций (1+1)-EA, за которые будет верно заполнен столбец t матрицы M не превышает

$$en\kappa m^2(1 + 1/2 + \ldots + 1/m) \leq en\kappa m^2(\ln m + 1).$$

Требуемая оценка на среднее число итераций до получения матрицы M^* следует из необходимости заполнить все n столбцов. ■

Заметим, что на каждой итерации (1+1)-EA требуется обновить матрицу M, что требует времени $O(nm(\theta_1+\theta_2))$ в предположении, что случайный выбор локальной операции выполняется за константное время. Таким образом, среднее время получения искомой матрицы M^* посредством (1+1)-EA составляет $O(n^3\kappa m^3(\theta_1+\theta_2)(\ln m+1)+\theta_5)$ и оптимальное значение целевой функции задачи P

$$f^* = \min\{G(M_{rn}, r) : r \in B_2 \times \ldots, \times B_\beta\}$$

вычислимо с помощью (1+1)-EA в среднем за время $O(n^3\kappa m^3(\theta_1+\theta_2)(\ln m+1) + \theta_5 + m\theta_4)$.

Рассмотрим модифицированный оператор мутации с учетом областей определения функций $F \in \mathcal{F}$. Пусть праобраз $F^{-1}(Q_r) = \{(s'_1, r') : F(s'_1, r') \in Q(r)\}$ может быть вычислен для любого $r \in B_2 \times \ldots \times B_\beta$. Тогда определение случайной локальной операции может быть модифицировано таким образом, что после равновероятного выбора индексов r и k, а также функции $\Phi_{rk} \in \mathcal{F}$, элемент Γ_{rk} выбирается теперь с равномерным распределением из множества $\Phi^{-1}_{rk}(Q(r))$. Общее число всевозможных исходов такой случайной локальной операции есть $O(n\kappa m d_{\max})$, где

$$d_{\max} = \max\left\{|\Phi_{rk}^{-1}(Q(r))| : r \in B_2, \times \ldots, \times B_\beta\right\}.$$

Соответствующая модификация доказательства теоремы 1 с учетом меньшего числа всевозможных исходов локальной операции дает доказательство следующей теоремы.

Теорема 2 *Среднее число итераций многокритериального (1+1)-EA с модифицированным оператором мутации до получения матрицы* $M = M^*$ *есть*

$$O(n^2 \kappa m d_{\max} \ln m).$$

В алгоритмах ДП для задачи 0/1-рюкзак и задачи коммивояжера, приведенных в § 2, $d_{\max} = 1$. Следовательно, среднее число итераций до получения матрицы M^* с помощью многокритериального (1+1)-EA с модифицированным оператором мутации составляет $O(n^3 p_{\max} \ln(n p_{\max}))$ для задачи 0/1-рюкзак и $O(2^N N^4 \ln N)$ для задачи коммивояжера.

5 Приложение

Определение 2 *[1, 3] Задача NP-оптимизации* P *– это тройка* $P = (I, Sol, f_x)$*, где*

1. $I \subseteq \Sigma^*$ *– множество индивидуальных задач* P *распознаваемое за полиномиальное время.*

2. Для данной индивидуальной задачи $x \in I$, $Sol(x) \subseteq \Sigma^*$ *обозначает множество допустимых решений* x*. Для* $x, y \in \Sigma^*$ *ответ на вопрос о принадлежности* y *множеству* $Sol(x)$ *может быть дан за полиномиальное время, и существует такой полином* h*, что* $|y| \le h(|x|)$ *для всех* $x \in I$ *и* $y \in Sol(x)$*.*

3.Для $x \in I$ *и* $y \in Sol(x)$, $f_x(y)$ *– положительная целочисленная целевая функция, которая вычисляется за полиномиальное время.*

Список литературы

[1] Кочетов Ю.А., Пащенко М.Г., Плясунов А.В. О сложности локального поиска в задаче о p-медиане // Дискретный анализ и исследование операций. – 2005. – Серия 2, том 12, N 2. – с. 44—71.

[2] Растригин Л.А. Статистические методы поиска // М.: Наука. – 1968.

[3] Ausiello G., Protasi M. Local search, reducibility and approximability of NP-optimization problems // Information Processing Letters. – 1995. – V. 54, p. 73–79.

[4] Bellman R.E., Dreyfus S.E. Applied dynamic programming // Princeton University Press, 1962.

[5] Held, M., Karp, R. M. A dynamic programming approach to sequencing problems // Journal of the Society for Industrial and Applied Mathematics. – 1962. – Vol. 10, – p. 196–210.

[6] Rechenberg I. Evolutionsstrategie: Optimerung Technischer Systeme nach Prinzipen der Biologischen Evolution // Stuttgart: Formann-Holzboog Verlag. – 1973.

[7] Scharnow J., Tinnefeld K., Wegener I. The analysis of evolutionary algorithms on sorting and shortest paths problems // Journ. Math. Mod. and Alg. – 2004. – Vol. 3. – p. 349–366.

[8] Woeginger G.J. When does a dynamic programming formulation guarantee the existence of a fully polynomial time approximation scheme (FPTAS)? // INFORMS Journal on Computing. – 2000. – Vol. 12, Issue 1. pp. 57 – 74.