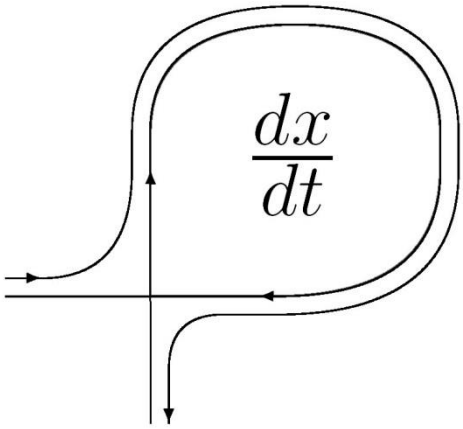

ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ И ПРОЦЕССЫ УПРАВЛЕНИЯ
N. 4, 2023
Электронный журнал,
рег. Эл № ФС77-39410 от 15.04.2010
ISSN 1817-2172

http://diffjournal.spbu.ru/
e-mail: jodiff@mail.ru

Информационные системы и процессы

Суммирование перечислителей в задачах дискретной оптимизации в контексте управления мастер-данными

Кузнецов С.В. [1,2,*]

[1]ООО «Юнидата»
[2]Санкт-Петербургский государственный университет

[*]sergey.kouznetsov@gmail.com

Аннотация. В данной работе рассматриваются субоптимальные задачи дискретной оптимизации. Формализована концепция перечислителей, предложенная известным ленинградским математиком И.В. Романовским. Представлен эффективный алгоритм перебора решений для суммы перечислителей на основе пересчёта границы Парето. Приводятся схема использования операции суммирования перечислителей при масштабировании решений задач по управлению мастер-данными.

Ключевые слова: перечислители, субоптимальные задачи, дискретная оптимизация, граница Парето, управление мастер-данными.

1. Введение

Задачи по оптимизации различных ресурсов активно изучались в Петербургской математической школе со времён Л.В. Канторовича [1]. В конце 90-х годов его ученик проф. И.В. Романовский предложил и развил концепцию процессов-перечислителей субоптимальных решений [2-4].

На сегодняшний день во многих практических задачах оптимизации требуется найти не только лучшее (оптимальное) решение, но k лучших решений, для которых целевая функция упорядочена по степени деградации. Такие задачи называются *субоптимальными* (реже используется термин k-*оптимальные*), и они часто возникают в комплексных задачах принятия решений, когда необходимо учитывать многочисленные факторы и рассматривать несколько «лучших решений». Субоптимальные задачи находят применение в различных бизнес-областях – при оценке рисков

[5], в управлении производством [6], в системах поддержки принятия решений [7,8], а также при управлении данными в биоинформатике и информационных системах [9].

Управление мастер-данными (Master Data Management) является особой областью управления данными крупных организаций [10], занимаясь агрегацией данных из различных источников и решением прикладных задач на основе этой агрегации. Данная область очень динамична, для неё создано большое количество методов, алгоритмов и готовых программных инструментов. В то же время в ней продолжает оставаться значительное количество проблем, связанных как со стоимостью и затратами, так и с эффективностью итоговых алгоритмов.

Перечислитель является процессом перебора решений некоторой субоптимальной задачи в порядке деградации целевой функции. Для перечислителей введен ряд операций – монотонное преобразование, слияние, суммирование и др. [2,3]. Эти операции полезны при масштабировании субоптимальных задач и способны понизить сложность итоговых алгоритмов, позволяя составлять их из них решения более простых задач (меньших по количеству допустимых решений, числу параметров оптимизации, объёму обрабатываемых данных и т.д.).

Среди операций над перечислителями особую роль играет суммирование (идея была предложена в [11]): эта операция позволяет объединять решения задач в разных доменах в искомое субоптимальное решение исходной задачи. В отличии от остальных операций для этой нетривиальным является составить алгоритм, перебирающий решения для суммы перечислителей. Конструированию такого алгоритма и доказательству его корректности посвящена данная работа.

Данная статья является обновленной версией статьи 2005 года [3]. Выполнено более строгое определение концепции перечислителя (в частности, рассмотрены бесконечные перечислители), предложена формализация алгоритма перебора решений для суммы перечислителей, исследованы дополнительные свойства данного алгоритма, а также предложена схема применения этих результатов при масштабировании задач управления мастер-данными [9,10]. Также по сравнению с исходной версией [3] был исправлен ряд ошибок.

Статья организована следующим образом. Представлена уточненная формальная концепция перечислителя, включая определение субоптимальной задачи дискретной оптимизации, доказан ряд свойств перечислителей (раздел 2). Представлена схема применения перечислителей и операции суммирования для масштабирования задач управления мастер-данными, дано строгое определение операции суммирования с вытекающей отсюда проблемой нахождения алгоритма перебора для итогового перечислителя-суммы (раздел 3). Формально введена граница Парето и доказан ряд её свойств (раздел 4). Предъявлен алгоритм перебора решений для суммы двух перечислителей, показано, как его можно распространить на сумму произвольного количества слагаемых, приводится формальное доказательство корректности алгоритма (раздел 5).

2. Субоптимальные задачи дискретной оптимизации и перечислители

В данном разделе кратко изложена и формализована концепция перечислителей решений субоптимальных задач дискретной оптимизации.

Определение 1. Пусть имеется пара (D, v), где D является конечным или счетным множеством некоторых объектов, а v – функцией вида $D \longrightarrow \mathbb{R}$, имеющей минимум на D ($\mathbb{R}$ – это множество вещественных чисел), называемая целевой функцией. Тогда задачей дискретной оптимизации будет поиск решения – элемента d^0 из D, на котором функция v достигает минимума, т.е. $d^0 = argmin_{d \in D}\, v\,(d)$. При этом D будем называть множеством допустимых решений данной задачи, а эту задачу дискретной оптимизации обозначим как $\overline{Dopt} = (D, v)$.

Переходя к рассмотрению субоптимальной задачи дискретной оптимизации отметим, что в литературе термин субоптимальный означает «не оптимальный, но близкий к оптимальному». Это легко формализуется в случае непрерывной целевой функции, например, в задачах оптимального управления [12]. В случае многоцелевой оптимизации субоптимальными решениями некоторой задачи называют те из них, которые оптимальны по одной из целевых функций [7]. Однако для

задачи дискретной оптимизации необходимо рассматривать различные способы оценки близости найденных решений к оптимальному [13].

Определение 2. Пусть имеется некоторая задача дискретной оптимизации $\overline{Dopt} = (D, v)$. Создадим на её основе *субоптимальную задачу дискретной оптимизации*, построив процесс P, который на шаге m возвращает следующее самое оптимальное (т.н. *субоптимальное*) решение задачи $\overline{Dopt}$ после решений $d^0, \dots, d^{m-1}$. При этом нас будут интересовать не все субоптимальные решения, а лишь первые k (поэтому субоптимальные задачи часто называют k-оптимальными); число k будем называть *значением субоптимальности*. Таким образом, саму задачу будем обозначать как $Dopt = (D, v, k, P)$.

Отметим, что k может быть бесконечным.

Нас интересует то, как может быть устроен процесс P, который мы будем называть далее перечислителем.

Определение 3. Пусть имеется некоторая субоптимальная задача дискретной оптимизации $Dopt = (D, v, k,\ P)$. Тогда процесс перебора P решений задачи $Dopt$, который далее мы будем называть *перечислителем*, зададим как $P = (\ d^0,\ T)$, где $d^0 \in D$ — оптимальное решение задачи $\overline{Dopt}$, а T является всюду определённой функцией $T\colon D \to D$ (т.н. *функцией перебора решений*), которая удовлетворяет следующим условиями:

$$\forall\ d \in D\colon v(d) \le v\big(T(d)\big) \tag{1}$$

$$\forall\ d' \in D\backslash\{d^0\}\ \exists!\ d'' \in D \mid T(d'') = d' \quad \text{и} \quad d'' \ne d' \tag{2}$$

$$\exists\ C \in \mathbb{R}\colon \forall\ d \in D \text{ либо } v > C, \text{ либо последовательность равенств } v(d) = v(T(d)) = \dots = v(T^p(d)) \text{ всегда конечна} \tag{3}$$

за исключением ситуации, когда $T(d) = d$.

Таким образом, функция T по уже имеющемуся решению задачи $Dopt$ выдаёт следующее в смысле оптимальности. При этом каждое следующее решение, выдаваемое T, не лучше предыдущего в смысле значения функции v, что задаётся условием (1).

Свойство (2) утверждает, что любой элемент из D, кроме начального, является образом некоторого другого элемента из D, полученного применением T. Единственность прообраза гарантирует отсутствие циклов в случае конечности D.

Свойство (3) выражает мысль о том, что D является множеством объектов, имеющих дискретную природу (например, множеством всех возможных путей в графе из 100 вершин), и даже если имеются, например у графов, вещественные веса на рёбрах, то эти значения рассматриваются с некоторой фиксированной точностью.

Лемма 1. Пусть имеется задача $Dopt$. Тогда её перечислитель $P = (d^0,\ T)$ позволяет упорядочить все элементы множества D следующим образом: $D = \{d^0, d^1 \dots\}$, $T(d^p) = d^{p+1}$, где $p \in [0, |D| - 1]$, если D конечно, или $p \in \mathbb{N}_0$, если D бесконечно.

Доказательство. В силу (2) для любого $d' \in D, d' \ne d^0$ можно построить последовательность $d^1, \dots d^i, \dots$ элементов из D такую, что $d' = d^1$, а $d^{i+1} = T^{-1}(\ d^i)$. В силу (1) имеем $v\big(\ d^{i+1}\big) \le v\ (\ d^i)$. Поскольку функция v имеет минимум, а также в силу (3), мы должны остановится в смысле уменьшения функции v. Мы не можем это сделать за счет зацикливания вида $T^{-1}(d) = d$ для одного из элементов нашей последовательности в силу (2). В силу (3) мы не можем зациклиться из-за равенства значений v на разных элементах D. Также мы не можем остановиться, достигнув минимума v на $d \ne d^0$, т.к. из этого d, в силу (2) мы должны двинуться дальше, а уменьшаться дальше v не может, значит, у нас должна быть цепочка равенств $v(d) = v\big(T(d)\big) = \dots = v\big(T^p(d)\big)$, которая, в силу (3), конечна. Итак, остановиться в смысле уменьшения значения функции v мы может лишь попав в d^0 поскольку этот элемент не имеет прообраза в смысле T.

Таким образом, элемент d' оказывается достижим из d^0 с помощью T, т.е. $\exists\, p \in \mathbb{N}: d' = T^{-p}(d^0)$. Далее, если у нас есть два произвольных элемента d' и $d'' \in D$, то оба они достижимы из d^0. Но с помощью T можно построить единственную траекторию из d^0, поэтому существует $n \in N$ такое, что либо $d' = T^n(d'')$, либо наоборот. Более того, очевидно, что на эту траекторию через конечное количество шагов попадёт любой элемент из D ■

Несложно доказать, что существует не более одного элемента $d \in D: T(d) = d$. В самом деле, если имеется два таких элемента, то, повторяя рассуждение, представленное в доказательстве леммы 1, оба они должны лежать на одной траектории, ведущей из d^0, но ни один из них не может предшествовать другому, поскольку из каждого не возможен путь далее по множеству D. Будем обозначать такой элемент как d^{fin}.

Очевидно также, что в случае конечности D такой элемент всегда должен иметься, поскольку T всюду определена. Таким образом, признаком окончания работы T в случае конечности D является зацикливание T на одном элементе (зацикливание на нескольких элементах, т.е. петля более чем из одного элемента) невозможно в силу единственности прообраза (см. условие (2)) – в месте склейки петли у соответствующего элемента тогда было бы два прообраза.

Кроме того, ясно также, что в этот элемент мы попадаем, когда всё множество D уже пройдено. Кроме того, такого элемента не может существовать, если D бесконечно.

Бесконечность множества D оправдана, поскольку существуют, например, локально-конечные графы, на которых можно решать различные субоптимальные задачи дискретной оптимизации. Также в практических задачах пространство решений может быть бесконечным за счет его постоянной перестройки – например, граф, в котором мы ищем кратчайший путь, может меняться. Однако на практике мы, как правило, имеем дело с конечным случаем, а при сильном изменении исходных структур данных целесообразно начинать вычисление T заново, таким образом получая локально-конечную задачу.

Далее в этой работе мы будем полагать множество D конечным, причём $|D| \geq k$ (мы оставляем случай бесконечного D для дальнейших исследований). Для конечного случая свойство (3) тривиально выполняется.

3. Масштабирование субоптимальных задач управления мастер-данными

Многие организации, имея разнообразные данные, хранят их раздельно, в разных форматах, СУБД и т.д. Также, зачастую с этими данными работают разные подразделения организации. Эффективное управление гетерогенными данными является важной чертой современного цифрового пространства крупных организаций. Очищенные, консолидированные, полные данные предприятий и организаций, извлечённые из различных источников (в том числе различных информационных систем), принято называть *мастер-данными* (master data) [10]. Значительное количество задач, с которыми сталкиваются крупные организации, может быть решено на основе этих данных – далее такие задачи будем называть *задачами управления мастер-данными*.

В задачах управления мастер-данными часто появляются субоптимальные задачи дискретной оптимизации. K-оптимальные решения таких задач требуются для обеспечения быстрой реакции системы (например, трейдерские задачи), для различных видов стоимостного анализа и тому подобных [11]. При этом оказывается, что в силу гетерогенности мастер-данных возникающие прикладные задачи эффективнее решать, разбив данные на домены [9], а итоговое решение составлять из решения этих, так называемых, *локальных* задач. Таким образом, мы приходим к необходимости масштабирования прикладных задач и соответствующих решений.

В том случае, когда нам нужно решить субоптимальную задачу дискретной оптимизации для сложного, составного домена, целесообразно и задачу, и данные разбить на части, решая локальные задачи, а общее решение составить как некоторую композицию решений локальных задач. Например, имея задачу поиска оптимальной конфигурации деталей для сборки автомобиля, мы можем решать её отдельно для ходовой части, для кузова и других крупных узлов, а затем составим общую оптимальную конфигурацию. Таким образом, мы масштабируем только данные

– сама же задача остаётся неизменной. При этом может оказаться, что в каждой части задача оптимальной конфигурации может иметь свою специфику – следовательно, и задачи могут оказаться разными.

Масштабирование задачи означает, что её можно выразить (математически) через локальные задачи. Фактически, всё сводится к операциям над целевыми функциями. При этом, если ограничиться простыми операциями, то не сложно составить итоговую целевую функцию на основе локальных. Однако может оказаться, что при этом создать итоговый перечислитель на основе локальных будет не так просто.

Говоря неформально, мультидоменная задача дискретной оптимизации в области управления мастер-данных при разбиении на домены может быть представлена (масштабирована) как сумма локальных задач, т.е. целевая функция здесь будет суммой локальных целевых функций. А как быть с перечислителем?

Простой пример показывает, что итоговый перечислитель не будет простой суммой локальных. Пусть необходимо простроить последовательность путей передвижения из точки C в точку M, неубывающую по их стоимости, начиная с самого дешевого, при условии, что эти точки находятся в разных городах. Очевидно, что каждый такой путь можно разбить на три составляющие (отрезки): (a) путь внутри города-отправления, начинающийся из точки C; (b) путь между городами; (c) путь внутри города-прибытия, заканчивающийся в точке M. Для каждого из этих отрезков построим перечислители субоптимальных путей, и тогда их сумма должна быть перечислителем путей исходной задачи. Если предположить, что стоимость пути между городами (отрезок (b)) существенно превышает стоимости путей внутри городов (отрезки (a) и (c)), то несколько первых оптимальных решений всей задачи будут иметь неизменный отрезок пути (b).

Пусть задан набор задач $\{Dopt_1, .., Dopt_n\}$, где $P_i = \left(T_i, {d_i}^0\right)$. Результатом операции суммирования этих задач является субоптимальная задача $DoptSum_s = Dopt_1 + \ldots + Dopt_n$ такая, что:

$$
\begin{aligned}
&DoptSum = \left(D_{sum}, v_{sum}, s, SUM(P_1, .., P_n)\right), \text{ где} \\
&D_{sum} = D_1 \times .. \times D_n,\ s = \prod_{i=1..n} k_i, \\
&\forall\ d = (d_1, \ldots d_n)\ \ v_{sum}(d) = \sum_{i \in 1:n} v_i(d_i), \\
&SUM(P_1, .., P_n) = (d^0,\ TSum),\ d^0 = \left\{{d_1}^0, .., {d_n}^0\right\}.
\end{aligned}
\tag{4}
$$

Для завершения определения операции суммирования в части результирующего перечислителя $SUM(P_1, .., P_n)$ нам нужно предъявить функцию перебора $TSum$, доказав, что она удовлетворяет условиям (1) и (2). Как мы отметили выше, функцию перебора результирующего перечислителя не получается задать простой формулой от $T_1, \ldots, T_n$. Однако интуитивно понятно, что такая функция существует, так как множество решений задачи $DoptSum$ конечно и является полностью упорядоченным.

Замечание 1. Для каждой субоптимальной задачи дискретной оптимизации существует единственная, с точностью до порядка следования решений с одинаковым значением целевой функции, функция перебора субоптимальных решений.

Это замечание непосредственно следует из определения 2, которое требует, чтобы субоптимальное решение d^m с номером m было оптимальным решением соответствующей задачи на множестве $D \backslash \{d^0, \ldots, d^{m-1}\}$.

4. Граница Парето

Для эффективной реализации функции перечисления субоптимальных решений для суммы перечислителей нам понадобится понятие *границы Парето*. Это понятие было введено в экономике и социологии В. Парето в начале XX века и означает множество точек в некотором пространстве, оптимальных по определённым параметрам, и для точек этого множества нельзя

улучшить один из параметров, не ухудшив другие. В настоящее время это понятие используются в различных отраслях экономики, теории принятия решений, управления производством и пр.

Рассматривая задачу $Dopt_{SUM}$ и сумму перечислителей $SUM(P_1,..,P_n)$, введём для каждого перечислителя-слагаемого P_j отрезок натурального ряда $I_j = 1{:}\,k_j$, где k_j – значение субоптимальности для задачи $Dopt_j$. Пусть $R = I_1 \times ... \times I_n$. Каждый элемент R однозначно определяет решение субоптимальной задачи $Dopt_{SUM}$ как сумму решений задач-слагаемых.

Определение 4. Набор $I = (i_1, \dots, i_n) \in R$ доминирует набор $J = (j_1, \dots, j_n) \in R$, если $i_p \le j_p$ для каждого $p \in 1{:}\,n$.

Таким образом, если рассматриваемые наборы индексов соответствуют порядковым номерам решений субоптимальных задач $Dopt_1 \dots Dopt_n$, то доминирующий набор I соответствует более оптимальным решениям, чем J, поэтому он и доминирует над J. Нетрудно убедиться в том, что доминирование является отношением частичного порядка на множестве картежей из R.

Лемма 2. Пусть набор $I = (i_1, \dots, i_n)$ доминирует набор $J = (j_1, \dots, j_n)$ и $DoptSum = Dopt_1 + \dots + Dopt_n$. Рассмотрим следующие решения задачи $DoptSum$: $d = \{{d_1}^{i_1}, .., {d_n}^{i_n}\}$ и $d' = \{{d_1}^{j_1}, .., {d_n}^{j_n}\}$, где ${d_p}^m$ – это m-ое субоптимальное решение задачи $Dopt_p$. В этом случае справедливо следующее: $v_{sum}(d) \le v_{sum}(d')$.

Доказательство. В силу того, что I доминирует J, а также условия (1), справедливо следующее неравенство:

$$\sum_{p \in 1:n} v_p(d_p^{i_p}) \le \sum_{p \in 1:n} v_p(d_p^{j_p}). \tag{5}$$

Из этого напрямую следует утверждение леммы ■

Определение 5. Пусть задано некоторое частично упорядоченное по отношению доминирования множество S (это означает, что для некоторых элементов s_1 и $s_2 \in S$ справедливо, что либо $s_1 = s_2$, либо s_1 доминирует s_2, либо s_2 доминирует s_1). *Границей Парето* множества S называется такое его подмножество B, что любой элемент $s \in S \setminus B$ доминируется каким-либо элементом из B, и никакой элемент из B не доминируется другим элементом из B.

5. Функция перебора решений для суммы двух перечислителей

Предъявим алгоритм для вычисления функции перебора $TSum$ задачи $DoptSum$ в случае двух слагаемых.

Для некоторой задачи $Dopt$ будем обозначать её i-е решение как $d^i = T^i(d^0)$. Логично предполагать, что уже вычисленные решения задачи $Dopt$ сохраняются и потом не вычисляются повторно, поэтому к ним возможен прямой доступ по индексу $\{d^i, v^i\} = T.get(i)$, где v^i – значение целевой функции i-го решения.

Итак, имеем задачи $Dopt_1$ и $Dopt_2$, перечислители $P_1 = \left({d_1}^0, T_1\right)$ и $P_2 = \left({d_2}^0, T_2\right)$, а также соответствующие значения субоптимальности k_1 и k_2.

Введём целочисленный массив G^0 длинной $k_1 + 1$, где $G^0[0] = k_2$, $G^0[i] = 1$ для $i \in 1..k_1$, а также множество $B^0 = \{(1,1)\}$.

На листинге 1 представлен алгоритм, который по $T_1, T_2, G^{m-1}, B^{m-1}$ выдаёт $\overline{d^m}, G^m, B^m$.

1 **Вход:** $T_1, T_2, G^{m-1}, B^{m-1}$
2 $G^m = G^{m-1}, B^m = B^{m-1}$

3 **for** each $b \in B^m$
4 $\{v_1, d_1\} = T_1.\text{get}(b.x)$
5 $\{v_2, d_2\} = T_2.\text{get}(b.y)$
6 **if** $v^m > v_1 + v_2$ **then**
7 $v^m = v_1 + v_2$
8 $\overline{d^m} = \{d_1, d_2\}$
9 s^m = b
10 **if** $G^m[s^m.x] < k_2$, **then**
11 $G^m[s^m.x]$++ /* $G(s^m[0])$ стал равен $s^m.y + 1$
12 $B^m = B^m \setminus \{s^m\}$
13 **if** $(1 < s^m.x < k_1)$ & ($G^m[s^m.x - 1] - s^m.y) = 1$) **then**
14 $B^m \cup= \{s^m.\text{x} + 1,\ G^m[\ s^m.\text{x} + 1]) \}$
15 **elseif** $G^m[s^m.x - 1] > G^m[s^m.x]$ || $(s^m.x = 1)$ **then**
16 $B^m \cup= \{[s^m.x,\ G^m[(s^m.x)]\}$
17 **Выход:** $\overline{d^m}$, G^m, B^m

Листинг. 1. Алгоритм вычисления m-го решения суммы двух перечислителей

Докажем, что представленный алгоритм реализует $TSum$, т.е. $\overline{d^m}$ является m-тым субоптимальным решением d^m задачи $DoptSum$. Введём следующие обозначения:

$$S^0 = R,$$
$$S^m = S^{m-1} \setminus \{s^m\}, \quad (6)$$

где s^m – это набор индексов, по которым однозначным образом определяется $\overline{d^m}$, выдаваемое алгоритмом листинга 1 (строки 3–9).

Лемма 3. Множество B^m, выдаваемое алгоритмом листинга 1, является границей Парето для множества S^m при m: $0 \le m \le k_1 * k_2$.

Доказательство. Для начала покажем, что алгоритм листинга 1 строит множество B^m, которое можно задать следующим образом:

$$B^m = \{ (x', y') \mid G^m[x' - 1] > G^m[x'], y' = G^m[x'],\ x' \in 1{:}k_1\}. \quad (7)$$

Доказательство (7) проведем индукцией по m. Для B^0 (7) очевидно выполнено. Согласно предположению индукции будем полагать, что оно выполнено для B^{m-1}.

Приступая к построению множества B^m положим его равным множеству B^{m-1}(строка 2). Далее элемент $s^m = (x, y)$ удаляется из B^m (строка 12), массив G^m будет отличаться от G^{m-1} только в элементе x: $G^m[x] = G^{m-1}[x]$ +1 и становится равным $y + 1$ (строка 11).

Отметим, что в итоге в множество B^m войдет элемент $(x + 1, G^m[x + 1])$ (строка 14), поскольку значение $G^m[x]$ на шаге m увеличено на единицу, а согласно предположению индукции для B^{m-1} выполнено $G^m[x + 1] \le y < G^m[x]$, поскольку $(x, y) \in B^{m-1}$. При этом данный элемент уже мог содержаться в B^{m-1} , например если $|G^m[x] - G^m[x + 1]| > 1$.

Поскольку элемент (x, y) удален из B^m, то $G^m[x - 1] > y + 1$, а значит элемент $(x,\ y + 1)$ должен войти в B^m, поскольку $G^m[x] = y + 1$. Отметим, что $G^m[0] = k_2$, следовательно, для $x = 1$ всегда выполняется этот сценарий. Это условие и определяет вхождение элемента $(x, G^m[x])$ в B^m, что реализовано в строках 15-16.

Таким образом мы показали, что B^m удовлетворяет (7).

Перейдём к доказательству утверждения леммы и проведём его индукцией по m. Для множества S^0 это утверждение очевидно, поскольку очевидно, что $B^0 \subset S^0$. При этом B^0 состоит из одного элемента $(1, 1)$, который доминирует все другие элементы S^0.

Предположим, утверждение выполнено для множества S^{m-1} при $m > 0$. Множество S^m получается удалением из S^{m-1} элемента $s^m = (x, y)$, который входил в B^{m-1} (строки 3, 9). При этом элемент s^m также удаляется из B^m (строка 12), а в B^m добавляются только элементы из S^{m-1} (строки 14, 16), значит, $B^m \subset S^m$.

Рассмотрим значение $G^m[x-1]$. Возможны следующие случаи: $G^m[x-1] = y+1$ или $G^m[x-1] > y+1$. Случай $G^m[x-1] < y+1$ невозможен по построению G^m. При этом будем полагать, что $x > 1$, поскольку если $x = 1$, то в множество B^m на шаге m войдет элемент $(1, y+1)$, и таким образом, очевидно, что B^m будет границей Парето для множества S^m.

Случай 1. $G^m[x-1] = y+1$ (строка 13). Это означает, что элемент $(x,\ y+1)$, определяемый новым значением $G^m[x]$, не попадает в множество B^m в силу (7). Однако там содержится элемент $(x-1, y+1)$, который доминируют элементы множества $\{(x', y') |\ x - 1 \le x',\ y+1 \le y'\}$, и элемент $(x+1, G^m[x+1])$, который доминирует $\{(x', y') |\ x+1 \le x',\ G^m[x+1] \le y'\}$. Пересекая эти множества и принимая во внимание, что элемент $(x,\ y)$ уже исключен из B^m, а множество B^{m-1} было границей Парето согласно предположению индукции, мы видим, что $B^m = \{B^{m-1} \setminus \{(x, y)\}\} \cup \{(x+1, G^m[x+1])\}$ и является границей Парето множества S^m.

Случай 2. $G^m[x-1] > y+1$. Этот означает, что в множество B^m войдет новый элемент $(x,\ y+1)$ (строка 16). Осталось лишь проверить те элементы, которые им не доминируются, а именно $E = \{(x', y') |\ x \le x', y = y'\}$. Выше показано, что элемент $(x+1, G^m[x+1]) \in B^m$, следовательно, доминируются элементы $\{(x', y') |\ x + 1 \le x',\ G^m[x+1] \le y'\}$. Тогда, принимая во внимание, что $(x,\ y)$ уже исключен из S^m и выполнено $G^m[x+1] \le y$, получаем, что множество $E \setminus \{(x, y)\}$ доминируется элементом $(x+1, G^m[x+1]) \in B^m$. Таким образом и в этом случае B^m является границей Парето множества S^m ■

Теорема 1. Пусть $DoptSum = Dopt_1 + Dopt_2$. Тогда алгоритм с листинга 1 задает $TSum$, т.е. $\overline{d^m} = d^m$ и $SUM(P_1, P_2)$ является перечислителем.

Доказательство. Согласно определению 2 для того, чтобы $SUM(P_1, P_2)$ был перечислителем, необходимо, чтобы для $TSum$ выполнялись условия (1) и (2).

Для доказательства истинности условия (1) применим индукцию по m, а именно, что $TSum$ выбирает решение d^m с наименьшим значением функции $v_{sum}(d^m)$ из еще не выбранных ранее, т.е. из множества $D_{sum} \setminus \{d^0, \dots, d^{m-1}\}$. В силу леммы 2 решение с минимальным возможным значением целевой функции v_{sum} лежит на границе Парето множества возможных решений $D_{sum} \setminus \{d^0, \dots, d^{m-1}\}$ и однозначным образом задается индексами подчиненных перечислителей. Таким образом, для обеспечения условия (1) достаточно выбрать решение с минимальным значением функции v_{sum} из таких элементов множества D_{sum}, которые построены по индексам решений подчиненных перечислителей из множества B^m, а последнее, согласно Лемме 3, является границей Парето множества индексов S^m. Выбор решения с минимальным значением v_{sum} реализован в строках 3–9 листинга 1.

Условие (2) для $TSum$ выполняется в силу того, что $\forall\ d = \{d_1, d_2\} \in D \setminus \{d^0\}$ истинно, что $\exists\ d' = \{d'_1, d'_2\} \in D_{sum}$, где либо $d_1 = TSum(d'_1)$ либо $d_2 = TSum(d'_2)$. Это следует из выполнения условия (2) для T_1 и T_2, а значит, элемент d оказывается достижим из d' за счет конечного числа суперпозиций $TSum$ над значением d', кол-во которых не превышает $max(k_1, k_2)$■

Сложность вычисления $TSum$ для случая суммирования двух перечислителей является линейной от количества элементов в границе Парето на очередном шаге, поскольку перебираются лишь элементы текущей границы Парето (одномерный массив элементов), а не все возможные сочетания пар решений. Соответственно, в худшем случае перебираются все решения одного из подчиненных перечислителей, а пересчет вспомогательных структур ограничивается запросами к массиву G^m.

Теперь, имея полностью определенную задачу $DoptSum = Dopt_1 + Dopt_2$ для двух слагаемых, мы можем рассмотреть случай $n > 2$. Имеем $\sum_{i=1}^{n} Dopt_i = (\dots(Dopt_1 + Dopt_2) + Dopt_3) + \cdots + Dopt_n) \dots)$, следовательно, итоговый перечислитель задачи $DoptSum$ получается с помощью $n-1$ попарных сложений перечислителей, и мы можем $n-1$ раз применить представленный выше алгоритм для вычисления очередного субоптимального решения задачи $DoptSum$ (т.е. функции $TSum$). А в силу замечания 1 мы имеем единственную функцию перебора субоптимальных решений для этой задачи.

Однако следует отметить, что может существовать больше одного алгоритм вычисления $TSum$ перебора несмотря на то, что все они будут иметь одинаковую (с точностью до перестановок равных субоптимальных решений) выдачу. Следует ожидать, что эти алгоритмы могут иметь разную сложность.

6. Заключение

Выше приведены формальные определения перечислителей субоптимальных решений в задачах дискретной оптимизации, а также операции их суммирования. Полученные результаты обосновывают эффективное использование теории перечислителей для задач поиска субоптимальных решений в различных областях, в том числе в области управления мастер-данными. Масштабирование субоптимальных задач и представленный алгоритм суммирования перечислителей позволяет существенно снизить сложность разработки алгоритмов, в частности, для задач управления данными и мастер-данными. Для рассмотренного выше примера известной задачи об оптимизации набора комплектующих для сложного изделия в условиях ценовой дисперсии и ограничения поставок, которая сводится к известной задаче о переборе k-кратчайших путей в графе [11], использование перечислителей и предложенного алгоритма их суммирования может позволить существенно снизить сложность решения исходной задачи [14], что, однако, является предметом дальнейшего исследования. Также небезынтересным является использование модельно-ориентированных решений [15, 16] для визуализации отдельных аспектов задачи управления мастер-данными в контексте данных исследований.

Благодарности

Хочу посвятить эту работу моему учителю профессору И. В. Романовскому, и выразить благодарность моему научному руководителю профессору Д. В. Кознову за помощь и внимание к деталям.

Литература

[1] Канторович Л.В. Математические методы организации и планирования производства. Л.: Изд-во ЛГУ, 1939, 68 с.

[2] Романовский И.В. Субоптимальные решения. Петрозаводск: Изд-во Петрозаводского университета. 1998.

[3] Романовский И. В., Кузнецов С. В. Обобщенный алгоритм суммирования перечислителей субоптимальных решений //Вестник Санкт-Петербургского университета. Математика. Механика. Астрономия. – 2005. – №. 2. – С. 74-87.

[4] Романовский И. В. Перебор субоптимальных решений в дискретных задачах оптимизации //Компьютерные инструменты в образовании. – 2012. – №. 6. – С. 25-34.

[5] Брумштейн Ю. М., Тарков Д. А., Дюдиков И. А. Анализ моделей и методов выбора оптимальных совокупностей решений для задач планирования в условиях ресурсных ограничений и рисков //Прикаспийский журнал: управление и высокие технологии. – 2013. – №. 3. – С. 169-180.

[6] Лукичева Н. М. О математических подходах в планировании, прогнозировании и управлении //Актуальные вопросы развития современного общества. – 2019. – С. 306-309.
[7] Кириллов Ю. В. Многокритериальное моделирование как основа информационных технологий поддержки принятия решений // Фундаментальные исследования. – 2004. – №. 6. – С. 85-97.
[8] Marler R. T., Arora J. S. Survey of multi-objective optimization methods for engineering //Structural and multidisciplinary optimization. – 2004. – Т. 26. – С. 369-395.
[9] Кузнецов С., Константинов А., Скворцов Н. Ценность Ваших Данных, Изд-во Альпина PRO, 2022.
[10] DAMA-DMBOK: Свод знаний по управлению данными. Второе издание. Москва: Олимп-Бизнес, 2020.
[11] Minieka E., Shier D. A note on an algebra for the k best routes in a network// Journ. Inst. Math. Appl. 1973. Vol. 11. P. 145–149.
[12] О. Е. Майкова. Субоптимальные режимы в задаче Фуллера // Труды МИАН, 2002, том 236, 226–229.
[13] Бабаев Д. А., Мамедов К. Ш., Мехтиев М. Г. Методы построения субоптимальных решений многомерной задачи о ранце //Журнал вычислительной математики и математической физики. – 1978. – Т. 18. – №. 6. – С. 1443-1453.
[14] Mohanta K. et al. Comprehensive study on computational methods for k-shortest paths problem // International Journal of Computer Applications. – 2012. – Т. 40. – №. 14. – С. 22-26.
[15] Кознов Д.В. Методология и инструментарий предметно-ориентированного моделирования. Диссертация на соискание ученой степени доктора технических наук / Санкт-Петербургский государственный университет. Санкт-Петербург, 2016.
[16] Кознов Д.В., Перегудов А.Ф., Бугайченко Д.Ю., Чернятчик Р.И., Казакова А.С., Павлинов А.А. Визуальная среда проектирования систем телевизионного вещания. Системное программирование. 2006. Т. 2. № 1. С. 142-168.

Summation of the enumerators in discrete optimization problems in the context of master data management

Kuznetsov S.V. [1,2,*]

[1] Unidata LLC
[2] Saint-Petersburg State University

[*]sergey.kouznetsov@gmail.com

Abstract. The publication presents an approach to the use of discrete optimization algorithms, in particular, the search for suboptimal solutions. The theory of enumerators, proposed by the famous Leningrad mathematician I.V. Romanovsky, and the operation of their summation, which is proposed to be used to create multi-domain suboptimal algorithms, are considered. The paper presents an efficient algorithm to sum enumerators based on the recalculation of the Pareto boundary. Motivations for using the proposed algorithm within the framework of a well–known task in the field of Master Data Management are given.

Keywords: enumerators, suboptimal problems, discrete optimization, Pareto boundary, master data management.

Acknowledgements I would like to dedicate this work to my teacher, Prof. I.V. Romanovsky, and thank my scientific supervisor, Prof. D.V. Koznov, for his help and attention to detail.