Содержание

Аннотация

В практических инженерных и оптимизационных приложениях решение задач многоцелевой оптимизации (МО) часто подразумевает использование методов скаляризации – сведение задачи МО к задаче с одной целевой функцией. Хоть известные подходы могут и быть эффективными, они часто сопряжены со значительными вычислительными затратами из-за необходимости итеративных вычислений, а также их использование усложняется необходимостью подбора гиперпараметров. В этой работе предлагается расширить определение конкурентного решения и построить на его основе метод скаляризации многокритериальной задачи. Предложенный метод имеет хорошую интерпретируемость и не требует настройки гиперпараметров. Для предложенного метода скаляризации предлагается способ получения приближенного решения в случае, когда функции липшицевы и их вычисление возможно только один раз. Это актуально, когда вычисления очень дорогие или повторное вычисление невозможно. Вычислительные эксперименты, проведенные для задачи конкурирующих потоков минимальной цены, демонстрируют работоспособность и масштабируемость предложенного подхода, подчеркивая его потенциал для решения вычислительных проблем в МО в различных областях.

1 Введение

Прикладные задачи часто формулируются так, что для них нет одного оптимального решения, и необходимо руководствоваться несколькими критериями. Для решения подобных задач есть методы многокритериального принятия решений и методы многокритериальной оптимизации [1]. Задачи из этой области и методы их решения находят широкое применение в различны областях. Примеры находятся в задачах выбора параметров в сети электропитания и телекоммуникаций [2, 3, 4, 5], задачах машинного обучения [6, 7, 8], химии [9], биологии [10], и задачах из инженерных областей [11]. В многокритериальной оптимизации распространено два подхода – аппроксимация парето фронта и скаляризация задачи, то есть сведение задачи к задаче с одним критерием оптимальности.

Для аппроксимации парето фронта существует множество методов: методы на основе генетических и эволюционных подходов [12, 13]. Такие алгоритмы не эффективны по количеству семплов и являются вычислительно дорогими. Интерактивные подходы [14]. Как и сказано в названии класса методов, для их использования требуется участие внешнего эксперта – человека. Это мешает широкому распространению подобных методов. Появляются работы основанные на байесовском подходе [15, 16] и на восстановлении направлений убывания функций [17]. Авторы байесовского подхода отмечают значительный прирост эффективности алгоритма по семплам по сравнению с предшествующими алгоритмами.

Методы скаляризации – сведения задачи к задаче одноцелевой оптимизации также широко распространены. Популярными являются методы взвешивания: линейное взвешивание, взвешенная t-ая степень, взвешенная квадратичная задача, ϵ ограничивающий подход. Семейство методов с целевой точкой: метод на основе функции расстояний, функции достижимости и другие. Семейство методов, основанных на направлениях: подходы Пасколетти, Серафини, Ф. Гембички(F Gembicki) и остальные. Названые методы в подробностях разобраны в книге [18]. Множество книг посвящено теме многокритериальной

оптимизации [1, 18, 19].

Методы восстановления парето фронта используются для подробного изучения оптимальных параметров задачи. Однако для современных задач подобное удовольствие слишком дорогое. Поэтому представляют интерес методы, позволяющие быстро находить набор решений с определенным свойством. По сути методы скаляризации ставят задачу такого поиска. Например, методы взвешивания задают приоритет на оптимизируемых функциях. Однако, по мере изучения методов скаляризации становится ясно, что методы требуют введения необучаемых параметров. В методах взвешивания такими параметрами являются веса, с которыми берутся функции. От этих параметров зависит сложность поиска решения и интерпретация полученного решения. Из-за свободы в выборе параметров у решений интерпретируемость низкая – они не обладают достаточно наглядным и понятным свойством. Эту неопределенность помогает решить идея конкурентного (competitive) решения. Она дает хорошо интерпретируемое на практике решение без введения большого количества параметров. Однако и у нее есть свои недостатки. В этой работе предлагается расширить определение конкурентного решения и на основе этого расширения поставить задачу оптимизации – метод скаляризации. Для липшицевых функций предлагается вычислительно эффективный метод поиска приближенного конкурентного решения. Метод предполагается использовать в случае сильного ограничения в вычислительных ресурсах и когда нет возможности повторно вычислять функции. Итеративная оптимизация в обоих случаях недоступна. Это актуально, поскольку современные задачи имеют большие размерности и градиентные и эволюционные методы для них неэффективны или даже неприменимы.

Дальнейший текст составлен следующим образом: в разделе 2 определяется задача оптимизации, вводится определение конкурентного решения и приводится обобщение определения. На основе нового определения ставится задача оптимизации и отмечается связь этой постановки с работами ранее. В разделе 3 вводится предлагаемый метод решения задачи для липшицевых функций и в разделе 4 доказываются некоторые утверждения для

предложенных методов. В разделе 5 приводятся численные эксперименты для демонстрации работы метода.

2 Постановка задачи

Задача многокритериальной оптимизации формулируется в следующем виде:

$$\min_x f \triangleq (f_1(x), ..., f_m(x))^T, \qquad (T_0)$$
$$\text{s.t. } x \in K.$$

Здесь $x \in \mathbb{R}^n$. Набор целевых функций $f_i : \mathbb{R}^n \to \mathbb{R}_{++} \quad i = \overline{1,m}$. Допустимое множество K рассматривается выпуклое, непустое и компактное. Например, подходит множество с линейными ограничениями вида $K = \{x \in \mathbb{R}^n : \ Ax \leq b\}$. Также K оснащено нормой $\|\cdot\| : K \to \mathbb{R}$.

Остается определить, что понимается под минимальностью, ведь в задаче дана вектор-функция. Один подход – использовать Парето оптимальность. Однако Парето оптимальных точек может быть бесконечно много. Нам необходимо получить одно решение, которое будет удовлетворять заданному хорошо интерпретируемому свойству. Далее предполагаем, что функции имеют положительные значения и их нужно минимизировать. Такими являются, например, функции, отражающие траты на производство. Рассмотрим определение конкурентного решения:

Определение 1 (Гамма конкурентное решение (старое определение))**.** *Пусть* $x_i = arg\min_x f(x)$*. Обозначим значение функции в этой точке* $f_i^* = f_i(x_i)$*. Тогда точка* x *называется* γ*-конкурентным решением для набора функций* f_i *если* $\forall i = \overline{i,m}$*:*

$$f(x) \leq (1+\gamma) f_i^*.$$

Это классическое определение конкурентного решения. Приведем пример, демонстрирующий недостаток такого подхода:

Комментарий 1. *Пусть компания работала в течение нескольких периодов с разными стратегиями. Стратегия задается точкой* x_i*. Для этих точек были посчитаны метрики* $f_k(x_i)$*. Необходимо выбрать стратегию для следующего периода. Компания не знает, как может развиться ситуация на*

рынке. Однако, она может выбрать такую стратегию, которая бы показывала результаты не сильно хуже на уже известных сценариях развития ситуации.

В данном случае мы не можем использовать приведенное выше определение конкурентности, поскольку у нас нет информации об оптимальности действий компании в исторических данных. Поэтому в новом определении конкурентности требование оптимальности в сравниваемых точках исключается:

Определение 2 (γ-конкурентное решение)**.** *Пусть* $x_i : f_i(x_i) = v_i$*. Тогда точка* x *называется* γ*-конкурентным решением для функций* f_i *в точках* x_i *если* $\forall i = \overline{i,m}$*:*

$$f(x) \leq (1+\gamma)v_i \tag{1}$$

Введем определение оптимального решения на основе этого свойства. Конкурентное решение с показателем γ это "достаточно хорошее"решение с точки зрения нашего нынешнего понимания значений целевых функций. Это определение перекликается с тем, как ставятся задачи на практике: даны исторические данные о том, как работала система, метрики посчитаны. Необходимо найти решение, которое удовлетворяет новому свойству и портит метрики как можно меньше. Чтобы записать задачу оптимизации сопоставим каждой функции f_i точку x_i. Обозначим $v_i = f_i(x_i)$. Ставим задачу:

$$\begin{aligned} &\min_{x,\gamma} \gamma, \qquad (T_1)\\ &\quad \text{s.t. } x \in K,\\ &\qquad f_i(x) \leq (1+\gamma)v_i, \quad \text{for} \quad i \in \overline{1,m}. \end{aligned}$$

Таким образом, цель оптимизации состоит в том, чтобы найти γ-конкурентную точку, которая удовлетворяет необходимым свойствам и дает лучший параметр γ для известных заданных значений. В поставленной задаче нет дополнительных параметров, что делает ее более конкретной и интерпретируемой.

2.1 Связь с другими работами

Метод скаляризации, предложенный выше T_1 тесно связан с подходом, который предлагается в [20]. В работе ставится следующая задача оптимизации:

$$\begin{aligned} &\min_{x,\gamma} \gamma, \\ &\quad \text{s.t. } x \in K, \\ &\qquad f_i(x) - w_i\gamma \leq f_i^*, \quad \text{for} \quad i \in \overline{1,m}. \end{aligned} \tag{T_1}$$

Здесь f_i^* интерпретируются как целевые значения для оптимизируемых функций. Они не обязательно связаны с какой-то точкой, и могут быть взяты из других соображений. w_i – относительная важность изменения i ой функции. По сути w_i задает направление, в котором могут меняться функции. Данная задача сводится к нашей взятием параметров $w_i = f_i^* = v_i$. Основное отличие нашей постановки от этой – мы связываем задачу оптимизации со значениями функций в заданных точках и фиксируем параметры. Фиксация параметров дает интерпретируемость полученному решению. Выбор значений в конкретных точках будет использоваться для поиска приближенного решения для случая липшицевых функций, к чему мы и переходим.

2.2 Мотивация поиска приближенного решения

Определение 3 (Липшицева фукция)**.** *Функция* $f : \mathbb{R}^n \to \mathbb{R}+$ *называется липшицевой на* K *с нормой* $\|\cdot\|$*:*

$$\exists L > 0 : \ \forall x, y \in K : \ |f(x) - f(y)| \leq L\|x - y\|.$$

L *– константа Липшица.*

В случае, когда итеративный поиск оптимальной точки невозможен или крайне трудозатратен, мы предлагаем метод приближенного решения поставленной задачи для липшицевых функций. Подобные задачи возникают, например в сетевой оптимизации: современные задачи из этой области имеют

большие размеры и могут быть выражены в виде задач линейного программирования [21, 22]. Согласно [23] задачи типа линейного программирования удовлетворяют условиям Липшица по параметрам b в ограничении $Ax \leq b$. Мы приводим пример такой задачи в разделе с экспериментами 5.

3 Предлагаемый метод аппроксимации

Соберем все обозначения вместе. Для каждого $i = \overline{1,m}$ дана функция $f_i(x) : \mathbb{R}^n \to \mathbb{R}_{++}$, которую необходимо минимизировать, Она липшицева с константами L_i и вычислена в точке $x_i : f_i(x_i) = v_i$. Необходимо найти $x \in K$ и γ, которые выполнимы для T_1. Как мы уже упоминали, метод не должен вычислять функции при поиске решения. К тому же, при сделанных предположениях T_1 не обязательно выпуклая. Остается воспользоваться только липшицевостью. Рассмотрим некоторые $x \in K$ и γ, для которых выполнено:

$$\|x_i - x\| \leq \frac{\gamma v_i}{L_i} \Rightarrow |v_i - f(x)| \leq \gamma v_i.$$

В силу липшицевости. Тогда выполняется одна из альтернатив:

1. $v_i >= f_i(x)$:
$$f_i(x) < (1+\gamma)v_i \tag{2}$$
2. $v_i >= f_i(x)$:
$$f_i(x) < v_i < (1+\gamma)v_i \tag{3}$$

То есть получаются условия из задачи T_1, и рассмотренные x, γ – выполнимые точки. А x – γ-конкурентная точка для заданных функций и точек. Введем аппроксимированную задачу оптимизации:

$$\begin{aligned} &\min_{x,\gamma} \gamma \\ &\text{s.t. } x \in K \\ &\quad \|x - x_i\| \leq \frac{1}{L_i}(\gamma v_i) \quad \forall i \in \overline{1,m} \end{aligned} \tag{T_2}$$

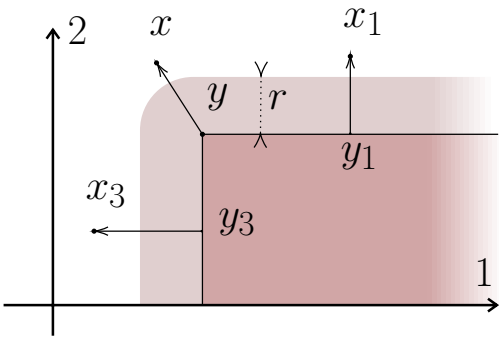

Рис. 1: Пусть функция f уменьшается по первому и увеличивается по второму параметру, и нужно ее минимизировать. Функция посчитана в точке y. В этом случае оператор возвращает $\mathbf{0}$ для точек в серой области. Для остальных точек оператор возвращает вектор, показанный на рисунке. Проекция выполняется по координатам. Затемненная область показывает область пространства, проекция из которой будет иметь норму, не превышающую r, указанную на рисунке.

Это выпуклая задача оптимизации, решение которого – γ-конкурентное решение на заданном множестве. Так как это приближение, полученное решение будет не лучше, чем точное решение задачи T_1.

Мы можем ослабить ограничения, если имеем информацию о монотонности функции по параметрам. Монотонность появляется, например, в задаче линейного программирования - функция монотонна по параметру b в ограничении $Ax \leq b$, поскольку их увеличение делает ограничения более мягкими. Для этого мы вводим $\texttt{clip}_f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ оператор для функции f:

$$\texttt{clip}_f(x, y)_i = \begin{cases} \max(x_i - y_i, 0), & f \text{ возрастает по } i\text{-ому параметру}, \\ \max(y_i - x_i, 0). & f \text{ убывает по } i\text{-ому параметру}, \\ x_i - y_i, & \text{иначе}. \end{cases} \tag{4}$$

Рисунок 1 демонстрирует работу оператора. Ставим задачу оптимизации, в которой разность заменена на оператор:

$$\min_{x,\gamma} \gamma, \tag{T_3}$$

$$\text{s.t. } x \in K,$$

$$\|clip_{f_i}(x,x_i)\| \leq \frac{1}{L_i}(\gamma v_i), \quad \forall i \in \overline{1,m}.$$

4 Теоремы

В этом разделе приведены результаты теоретического анализа приведенных задач оптимизации. Поставленные задачи являются выпуклыми. В теореме 1 показано, что решение задачи T_3 удовлетворяет необходимым условиям – а именно является выполнимой точкой для исходной задачи. В теореме 2 рассмотрено, какое решение можно получить, если знать только приближения констант Липщица. Такая ситуация распространена в приложениях из-за отсутствия полного доступа к функциям.

Теорема 1 (Latypov, 2024)**.** *Решение (x, γ) задачи T_3 является выполнимой точкой для T_1.*

Доказательство. Пусть $z_i = \texttt{clip}_{f_i}(x, x_i)$ и $y_i = x - z_i$. Для этого y_i выполняется $f_i(y_i) \leq f_i(x_i)$ в силу монотонности функции и способа построения точки y_i. Также воспользуемся неравенством $\|x - y_i\| = \|z_i\| \leq \frac{\gamma v_i}{L_i}$ и получим $f_i(x) - f_i(y_i) \leq \gamma v_i$. Просуммировав неравенства получаем требуемое утверждение: $f_i(x) - f_i(x_i) \leq \gamma v_i$. □

Теорема 2 (Latypov, 2024)**.** *Пусть даны аппроксимации констант Липшица для функций, обозначим их $\widetilde{L}_i = \kappa_i L_i$. Обозначим γ^* – решение задачи T_3. Пусть x – решение задачи $\widetilde{T}_3$:*

$$\begin{aligned} &\min_{x,\gamma} \gamma, \qquad (\widetilde{T}_3)\\ &\quad s.t.\ x \in K,\\ &\qquad \|clip_{f_i}(x,x_i)\| \leq \frac{1}{\widetilde{L}_i}(\gamma v_i), \quad \forall i \in \overline{1,m}. \end{aligned}$$

Для полученного решения x выполнено:

$$|f(x) - f(x_i)| \leq \frac{\kappa_{\max}}{\kappa_i}\gamma^* v_i$$

Здесь $\kappa_{\max} = \max_{i=\overline{1,m}} \kappa_i$.

Комментарий 2. *Если все константы умножены на один и тот же множитель, то мы получим тот же x что и для задачи с точными константами.*

Значительное ухудшение значения функции может произойти в случае, если какие-то константы оценены сильно хуже чем остальные.

$\widetilde{T_3}$ отличается от T_3, только заменой констант Липшица на приближенные.

Доказательство. Введем задачу оптимизации, в которой все константы Липшица домножены на κ_{max}:

$$\begin{aligned} &\min_{x,\gamma} \gamma, \\ &\quad \text{s.t. } x \in K, \\ &\qquad \|\texttt{clip}_{f_i}(x, x_i)\| \leq \frac{1}{\kappa_{\max} L_i}(\gamma v_i), \quad \forall i \in \overline{1,m}. \end{aligned} \tag{T_4}$$

Обозначим $\widetilde{\gamma}$ решение T_3 с константами $\widetilde{L_i}$, соответствующая ей задача $\widetilde{T_3}$. Также через $\overline{\gamma}$ – решение задачи T_4.

Заметим, что если в задачах T_3 и T_4 сделать замены $\gamma = a$ и $\frac{\gamma}{\kappa_{\max}} = a$ соответственно, то получим одну и ту же задачу, следовательно выполнено $\gamma^* = \frac{\overline{\gamma}}{\kappa_{\max}}$.

Для задач $\widetilde{T_3}$ и T_4 выполнено соотношение $\widetilde{\gamma} \leq \overline{\gamma}$: можем рассмотреть пересечение шаров в решении T_4 с параметром $\overline{\gamma}$. Если радиусы шаров увеличить то пересечение станет больше и сможем уменьшить γ, что и происходит в $\widetilde{T_3}$.

Итого получаем $\widetilde{\gamma} \leq \kappa_{\max}(\gamma^*)$ и из условий на радиусы в $\widetilde{T_3}$ x удовлетворяет условиям:

$$|f(x) - f(x_i)| \leq \frac{L_i}{L_i \kappa_i}(\widetilde{\gamma}) \leq \frac{\kappa_{\max}}{\kappa_i}(\gamma^*). \tag{5}$$

□

Комментарий 3 (Оптимальность полученной оценки)**.** *Для полученной оценки ухудшения качества приближенного решения рассмотрим пример с двумя функциями: $f_1(x) = 1 + L_1|x|$ и $f_2(x) = 1 + L_2|x - 1|$ и возьмем $x_1 = 0, x_2 = 1$*

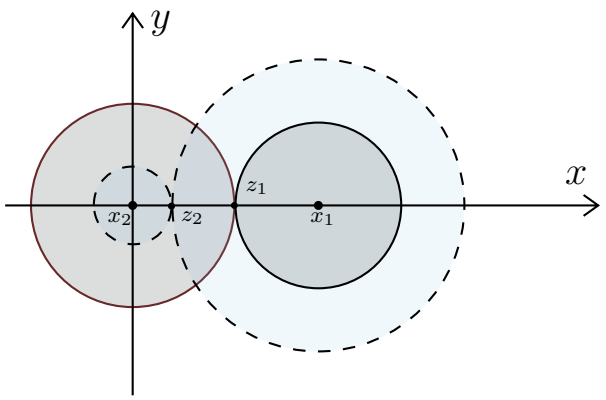

Рис. 2: Результат неточной оценки констант Липшица. Пересечение серых кругов z_1 – точное решение. Однако алгоритм находит z_2 из-за неправильной оценки констант.

Тогда для T_3 *получаем решение* $\gamma^* : (\gamma^*)(\frac{1}{L_1} + \frac{1}{L_2}) = 1$. *Для этой же задачи с неточными константами* $\widetilde{L}_i = \kappa_i L_i$ *получаем* $\widetilde{\gamma} : (\widetilde{\gamma})(\frac{1}{L_1\kappa_1} + \frac{1}{L_2\kappa_2}) = 1$.

Отсюда для случая $L_2 << L_1$ *и* $\kappa_1 = 1$*(точно знаем* L_1*):*

$$\frac{\widetilde{\gamma}}{\gamma^*} = \frac{\frac{1}{L_1} + \frac{1}{L_2}}{\frac{1}{L_1\kappa_1} + \frac{1}{L_2\kappa_2}} = \frac{\frac{L_2}{L_1} + 1}{\frac{L_2}{L_1\kappa_1} + \frac{1}{\kappa_2}} \approx \kappa_2 \tag{6}$$

То есть полученное решение будет близко к границе, полученной в теореме. Рисунок 2 наглядно показывает смещение точки, которую находит алгоритм.

5 Численные эксперименты

Для демонстрации работы предложенного метода было проведено два эксперимента. Первый эксперимент основан на сценарии классической задачи линейного программирования, а именно на задаче производства. Во втором эксперименте рассматривается задача конкурирующих потоков минимальной цены (Minimum Cost Concurrent Flow, MCCF). В этом эксперименте целевые функции имеют похожую структуру, но отличаются в параметрах, отвечающих за сценарий использования сети. Код экспериментов расположен по адресу https://github.com/intsystems/NIR_LatypovIM.

5.1 Задача производства

Идея эксперимента основана на классической задаче линейного программирования, но расширяет её. Пусть дана компания, деятельность которой можно разделить на периоды. Компания производит k видов продукции на сумму $x \in \mathbb{R}^k$ и реализует ее по стоимостям $c \in \mathbb{R}^k$. Для этого компании требуется n типов ресурсов со стоимостями $b \in \mathbb{R}^n$, которые приобретаются по цене $c_b \in \mathbb{R}^n$ в начале периода. Некоторые ресурсы могут закончиться в течение этого периода, поэтому компания может докупать недостающие ресурсы y во время периода по повышенной стоимости $c_a \in \mathbb{R}^n$: $c_a \geq c_b$ - покомпонентно. Для производства i-го продукта компания использует j-ый ресурс в количестве a_{ij}. Обозначим затраты ресурсов как матрицу $A = \|a_{i,j}\|_{i,j=1} \in \mathbb{R}^{n \times k}$.

Для упрощения выкладок и описания эксперимента предполагаем, что компания докупает недостающие ресурсы ровно столько, сколько ей нужно. Это реализуется, если компания заказывает малые порции ресурсов в покупках внутри периода.

В рассматриваемой постановке x – случайная величина, распределение которой неизвестно. Семплируется она один раз в период, так что собрать и исследовать большую выборку нет возможности. С учетом всех затрат, доход

компании за рассмотренный период дается следующим выражением:

$$f(b, x, c, c_b, c_a) = c^T x - c_b^T b - c_a^T y,$$
$$y = \max(0, Ax - b).$$

Эти функции Липшицевы, например в норме L_1 с константами

$$\max(\mathrm{abs}(c_b), \mathrm{abs}(c_a - c_b)).$$

Пусть компания проработала m периодов и пронаблюдала величины $i \in \overline{1, m} : c_i, x_i, c_b^i, c_a^i$ для начальных объемов ресурсов b_i. Необходимо найти стратегию приобретения ресурсов в начале периода, которая даст достаточно хороший доход в случае, если ситуация будет похожа на одну из предыдущих. Естественным ограничением здесь является ограничение по бюджету: $K = \{b : c_b^T b \leq B\}$. В эксперименте мы варьируем бюджет и получаем результаты.

5.1.1 Численные результаты

Рассматриваем количество случаев $m = 50$, количество видов ресурсов $n = 80$, количество видов продуктов $k = 30$. Варьируем бюджет B. Для поиска приближенного решения используем нормы $\|\cdot\|_1$, $\|\cdot\|_2$ с точными константами Липшица. Также рассматриваем $\|\cdot\|_2$ в которой все константы Липшица заменены на значение 100. Все результаты сравниваются с точным решением T_1.

На рисунке 3 отображено среднее качество решения для разных бюджетов B. Точка представляет собой среднее качество решения, полученного алгоритмом для разных функций. А именно рассматривается отношение значений функции в найденной точке к значениям функций в исходных точках. Рисуется их среднее значение и дисперсия. Эти значения отображены на оси y. По Оси x отложены бюджеты, для которых проводились вычисления. Слева направо изображены результаты для разных норм: $\|\cdot\|_1$; $\|\cdot\|_2$; $\|\cdot\|_1$ с неточными константами – все $\widetilde{L}_i = 100$. График справа представляет собой точное решение. Результаты визуально не отличимы в силу простоты рассмотренной задачи.

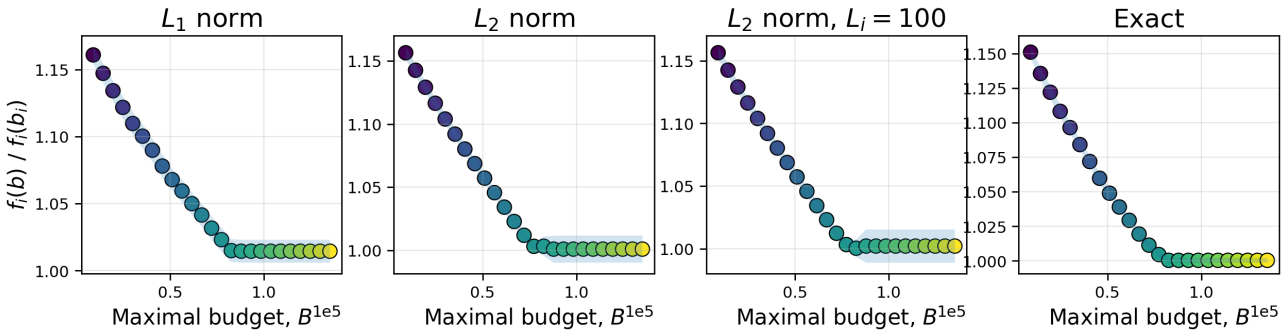

Рис. 3: Зависимость относительного прироста функции от бюджета для разных способов решения задачи производства.

Чтобы увидеть различия, рассмотрим график с относительными значениями качеств решений: рисунок 4. Из графика с точным решением вычитаем остальные графики и делим разность на точное решение для нормализации. Значение по оси y выше для более качественного решения. Видно, что алгоритмы на норме $\|\cdot\|_2$ работают одинаково и качество их решения отличается от точного не более чем на 1%. Норма $\|\cdot\|_1$ получает результаты хуже – качество решениия ухудшается до 2%. Объясняется это тем, что алгоритм получает более разреженное решение, что чаще приводит к нехватке некоторых ресурсов. Для всех норм видим ухудшение качества решения при некотором значении бюджета. Это случается из-за того, что алгоритм начинает закупать ресурсы, которые являются лишними для рассмотренных сценариев.

5.2 Конкурирующие потоки минимальной цены

Задача конкурирующих потоков минимальной цены (minimal cost multicommodity flow, MCF) ставится следующим образом: задан граф $\mathcal{G}(V, E)$, у которого m вершин и n ребер. У ребер $e \in E$ задана пропускная способность b_e и стоимость единицы потока c_e для этой пропускной способности. Кроме графа заданы потоки, которые необходимо обеспечивать сетью: (s_i, t_i, f_i). Эта тройка означает источник s, сток t и требуемый объем потока f. Обозначим эти запросы через матрицу корреспонденций D: $\forall i : D_{s_i,t_i} = f_i$. Требуется найти потоки в графе $F \in \mathbb{R}^{n \times m}$, которые обеспечат выполнение всех заказов при минимально

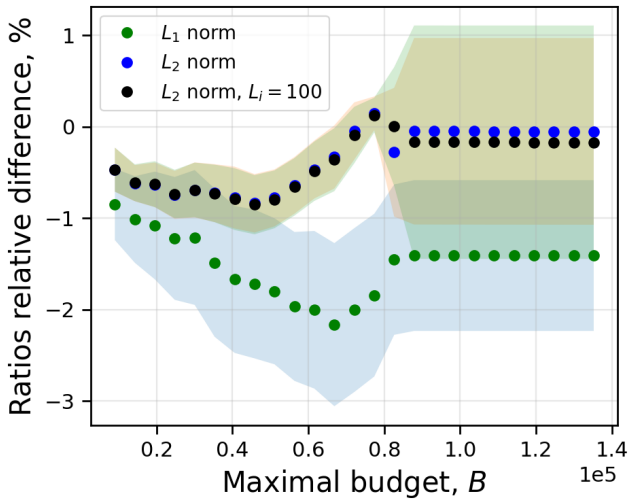

Рис. 4: Относительная разность качества приближенного решения с качеством точного решения для задачи производства.

возможной стоимости.

Задача переписывается в виде задачи линейного программирования [24]. Для этого определяем матрицу смежности вершин и ребер $A \in \mathbb{R}^{m \times n}$. Каждый столбец содержит ровно два ненулевых значения. Столбец, соответствующий ребру (i, j) содержит "$+1$"в строке i, "-1"в строке j, и нули во всех других строках. Для каждой вершины вводится вектор поставок $d_i \in \mathbb{R}^m$, он нужен для записи всех потоков, исходящих из этой вершины. При помощи матрицы корреспонденции D компоненты d_i расписываются в виде:

$$d_{ij} = \begin{cases} \sum_{k \neq j} D_{ik}, & i == j, \\ -D_{ij}, & i \neq j. \end{cases}$$

Соединим эти вектор-столбцы в матрицу и обозначим её $D_A \in \mathbb{R}^{m \times m}$, эта матрицы используется для удобной записи задачи. Используя [24], мы записываем оптимизационную задачу следующим образом:

$$\min_F c_e^T F\mathbf{1},$$
$$\text{s.t. } F\mathbf{1} \leq b,$$
$$AF = D_A.$$

в этой формулировке задача может не иметь выполнимой точки в силу ограничения с b. Чтобы убрать ограничения по b и сделать задачу проще добавим штраф y к задаче:

$$\min_F c^T F\mathbf{1} + c_a^T y, \qquad (MCCF)$$
$$\text{s.t. } F\mathbf{1} \leq b + y,$$
$$AF = D_A.$$

Этот штраф интерпретируется как аренда дополнительной пропускной способности в сети. Обозначим $g(b, D) = \text{value}(MCCF(b, D))$.

Теперь опишем сценарий использования метода. Есть некая компания, которая предоставляет услугу доставки посылок/сообщений в некой сети. Для работы ей необходимо арендовать полосу пропускания. Она может арендовать полосу пропускания в начале периода по цене c_b и арендовать дополнительную полосу пропускания в течение периода по цене c_a. На практике это происходит поэтапно, поскольку она арендуется заранее на длительный срок. В каждый период времени существуют требования на поставки – матрица корреспонденции D_i, потоки из которой необходимо выполнить. Эти матрицы задают сценарии использования сети. Компании они не известны на момент начала периода. Пусть она в течение m периодов работала с разными арендованными мощностями b_i и пронаблюдала D_i. В конце периода она подводит итоги и наблюдает свои расходы внутри периода $f_i(b_i) = g(b_i, D_i)$. Эти функции удовлетворяют условиям Липшица. Необходимо при заданном бюджете закупить пропускную способность так, чтобы она обеспечила небольшие траты

для разных ситуаций. Компании на данный момент известны только те D_i которые она пронаблюдала. Тут и появляется необходимость решать задачу T_1. Ограничения $K = \{b : c_b^T b \leq B\}$.

5.2.1 Численные результаты

В эксперименте используем топологию "germany50- из датасета SNDLib [25]. Это сеть из 50 вершин и 176 ребер. В датасете совместно с самим графом приводится набор матриц корреспонденции и стоимостей потоков в ребрах. Эти матрицы будут использованы в качестве матриц D. Для решаемости задачи при меньших пропускных способностях провели разреживание запросов: с вероятностью 0.4 ячейка в матрице обнуляется. Параметры для экспериментов генерировались следующим образом: стоимости потоков c есть в самих графах. На их основе генерировались начальная стоимость аренды и стоимость аренды внутри периода. Если пропускная способность стоит дешевле, то это более качественная пропускная полоса, поэтому стоимость такой полосы должна быть выше. Также считаем аренду дороже, чем обслуживание. Поэтому стоимости в начале периода генерировались как $(c_b)_i = (1/\sqrt{c_i})\xi$, где $\xi \sim U[9, 11]$. Стоимость аренды во время периода считаем выше: $(c_a)_i = (c_b)_i * \xi, \sim U[1.05, 1.15]$, то есть дороже от 5 до 15

Для этой задачи точное решение задачи найти не удалось, так как решатель не сошелся. Рассмотрели нормы $\|\cdot\|_1$; $\|\cdot\|_2$; $\|\cdot\|_\infty$. Рисунок 5 показывает относительный прирост значений функций при разных значениях бюджета. Точка это усреднение отношений значений функций к значениям в изначальных точках. Закрашенная область – $\pm$ дисперсия. По картинкам видно, что $\|\cdot\|_\infty$ работает хуже остальных.

Так как для этого случая нет точного решения, сравним эти решения с средним качеством решения. Это показано на рисунке 6. Больше значение – лучше. Видим, что разные нормы лучше работают для разных бюджетов: L_1 норма выдает разреженные решения, они не оптимальны для небольших бюджетов, поскольку некоторые ресурсы мало покупаются, но хорошо для

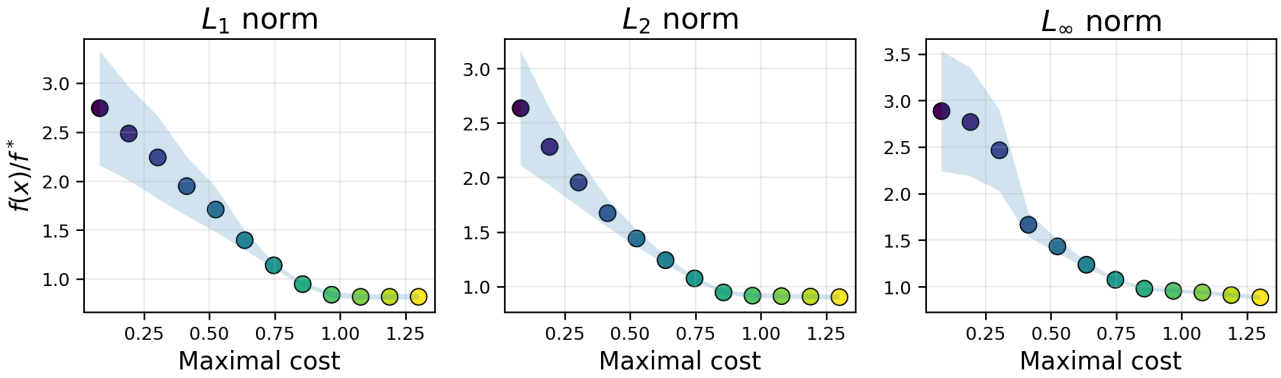

Рис. 5: Зависимость относительного прироста функции в зависимости от бюджета для разных способов решения задачи MCCF.

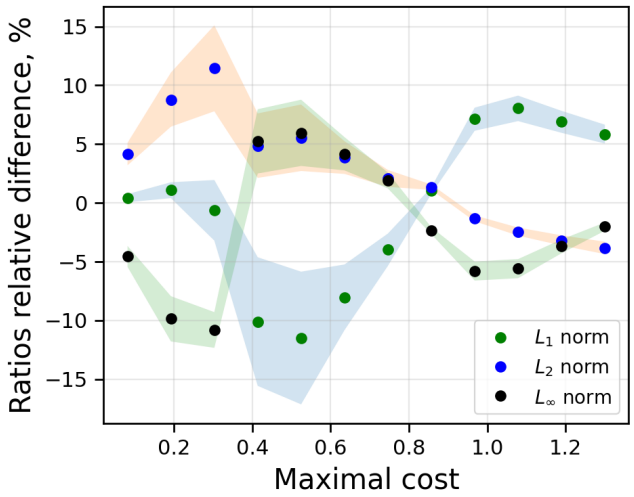

Рис. 6: Относительная разность качеств приближенного решения со средним качеством решения для задачи MCCF.

большого бюджета, поскольку лишние ресурсы покупаются самые дешевые. L_2 норма получает решение, которое до 10% лучше среднего качества решения. Аналогично L_1, но уже на больших бюджетах.

6 Выводы

В работе предложен метод скаляризации, не требующий подбора гиперпараметров. Он основан на обобщении определения конкурентного решения. Мы также показали связь нашего метода скаляризации с подходом, предложенным в [20]. Наш метод является частным случаем этого подхода и, благодаря фиксации параметров, его решения обладают хорошей интерпретацией – являются γ-конкурентными решениями.

Для липшицевых и липшицевых монотонных функций мы представили метод приближенного решения задачи оптимизации. Этот метод особенно полезен в случаях, когда вычисления функций являются дорогими, так как он работает только с заранее вычисленным набором значений. Необходимость таких методов обусловлена масштабами современных задач, например, в области оптимизации топологии телевизионных сетей и сетей поставок. Проведенные эксперименты продемонстрировали эффективность и практическую полезность предложенного метода, подтверждая его применимость в разных сценариях.

Список литературы

[1] M Murat Köksalan, Jyrki Wallenius, and Stanley Zionts. *Multiple criteria decision making: from early history to the 21st century*. World Scientific, 2011.

[2] Fulya Altiparmak, Mitsuo Gen, Lin Lin, and Turan Paksoy. A genetic algorithm approach for multi-objective optimization of supply chain networks. *Computers & industrial engineering*, 51(1):196–215, 2006.

[3] Mohammed Elmusrati, Hassan El-Sallabi, and Heikki Koivo. Applications of multi-objective optimization techniques in radio resource scheduling of cellular communication systems. *IEEE Transactions on Wireless Communications*, 7(1):343–353, 2008.

[4] Ernesto Mastrocinque, Baris Yuce, Alfredo Lambiase, and Michael S Packianather. A multi-objective optimization for supply chain network using the bees algorithm. *International Journal of Engineering Business Management*, 5:38, 2013.

[5] Emil Bjornson, Eduard Axel Jorswieck, Mérouane Debbah, and Bjorn Ottersten. Multiobjective signal processing optimization: The way to balance conflicting metrics in 5g systems. *IEEE Signal Processing Magazine*, 31(6):14–23, 2014.

[6] Thorsten Suttorp and Christian Igel. Multi-objective optimization of support vector machines. *Multi-objective machine learning*, pages 199–220, 2006.

[7] Marcela Zuluaga, Guillaume Sergent, Andreas Krause, and Markus Püschel. Active learning for multi-objective optimization. In *International conference on machine learning*, pages 462–470. PMLR, 2013.

[8] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

[9] Gade Pandu Rangaiah and AB Petriciolet. Multi-objective optimization in chemical engineering. *Developments and applications/edited by Gade Pandu Rangaiah, Adrián Bonilla-Petriciolet*, 2013.

[10] Yadira Boada, Gilberto Reynoso-Meza, Jesús Picó, and Alejandro Vignoni. Multi-objective optimization framework to obtain model-based guidelines for tuning biological synthetic devices: an adaptive network case. *BMC systems biology*, 10:1–19, 2016.

[11] R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26:369–395, 2004.

[12] Patrick Ngatchou, Anahita Zarei, and A El-Sharkawi. Pareto multi objective optimization. In *Proceedings of the 13th international conference on, intelligent systems application to power systems*, pages 84–91. IEEE, 2005.

[13] Abdullah Konak, David W Coit, and Alice E Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability engineering & system safety*, 91(9):992–1007, 2006.

[14] Kaisa Miettinen, Francisco Ruiz, and Andrzej P Wierzbicki. Introduction to multiobjective optimization: interactive approaches. In *Multiobjective optimization: interactive and evolutionary approaches*, pages 27–57. Springer, 2008.

[15] Shinya Suzuki, Shion Takeno, Tomoyuki Tamura, Kazuki Shitara, and Masayuki Karasuyama. Multi-objective bayesian optimization using pareto-frontier entropy. In *International conference on machine learning*, pages 9279–9288. PMLR, 2020.

[16] Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Multi-objective bayesian optimization over high-dimensional search spaces. In *Uncertainty in Artificial Intelligence*, pages 507–517. PMLR, 2022.

[17] Bennet Gebken and Sebastian Peitz. An efficient descent method for locally lipschitz multiobjective optimization problems. *Journal of Optimization Theory and Applications*, 188(3):696–723, 2021.

[18] S. Greco. *Multiple Criteria Decision Analysis: State of the Art Surveys*. International Series in Operations Research & Management Science. Springer New York, 2006.

[19] Kaisa Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 1999.

[20] F Gembicki and Y Haimes. Approach to performance and sensitivity multiobjective optimization: The goal attainment method. *IEEE Transactions on Automatic control*, 20(6):769–771, 1975.

[21] JC Moreno Banos and Markos Papageorgiou. A linear programming approach to large-scale linear optimal control problems. *IEEE transactions on automatic control*, 40(5):971–977, 1995.

[22] Richard Kipp Martin. *Large scale linear and integer optimization: a unified approach*. Springer Science & Business Media, 2012.

[23] Olvi L Mangasarian and T-H Shiau. Lipschitz continuity of solutions of linear inequalities, programs and complementarity problems. *SIAM Journal on Control and Optimization*, 25(3):583–595, 1987.

[24] Mokhtar S Bazaraa, John J Jarvis, and Hanif D Sherali. *Linear programming and network flows*. John Wiley & Sons, 2011.

[25] Sebastian Orlowski, Roland Wessäly, Michal Pióro, and Artur Tomaszewski. Sndlib 1.0—survivable network design library. *Networks: An International Journal*, 55(3):276–286, 2010.