ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

Отчет о программном проекте

на тему Стохастическое динамическое программирование

Выполнил:
студент группы БПМИ 2010 Ураков / Е. М. Ураков
Подпись / И.О. Фамилия

18.05.2022
Дата

Принял:
руководитель проекта Хамидов Олег Валерьевич
Имя, Отчество, Фамилия

заведующий отделом прикладной математики
Должность

институт систем энергетики им. Л.А. Мелентьева СО РАН
Место работы

Дата 18.05.2022 10
Оценка (по 10-тибалльной шкале) / Подпись

Москва 2022

Реферат

Объектом исследования данного проекта являются различные алгоритмы стохастического динамического программирования на примере многоэтапных задач с вероятностной неопределённостью разной степени сложности.

Целью проекта стало знакомство со стохастическим динамическим программированием, анализ алгоритмов, применяемых в этой области и их реализация.

Задачи проекта:

1. Изучить возможные способы решения задач методом стохастического динамического программирования.
2. Решить этими способами ряд учебных и, если получится, реальных задач на языке программирования Python.
3. Проанализировать эффективность различных подходов.
4. Составить пособие с примерами решения задач методом стохастического программирования.

Результат проекта:

Составлено пособие из 11 решённых задач с подробным описанием использованных математических моделей и анализом применяемых алгоритмов.

Содержание

Основные термины и определения

1. *Случайная величина* – переменная, значение которой не известно заранее, но для которой известно множество возможных значений и вероятностное распределение по этому множеству.

   Пример 1: подбрасывание монетки.
   Подбрасывается правильная монетка. Пусть X = 0, если выпал орёл. X = 1, если выпала решка. Тогда X – случайная величина, принимающая значение 0 с вероятностью ½ и значение 1 с вероятностью ½.

   Пример 2: бросание кубика.
   Кидается правильная игральная кость, каждая грань может выпасть с равной вероятностью. Пусть X – значение на кубике. Тогда X – случайная величина, принимающая значения из множества {1, 2, 3, 4, 5, 6}, причём каждое из своих возможных значений X принимает с вероятностью 1/6.

   Пример 3: равномерное распределение.
   Случайна величина X – произвольно выбранная точка на отрезке [a, b] вещественной координатной прямой.

2. *Математическое ожидание случайной величины* – среднее значение случайной величины. Говоря простым языком, если представить, что случайная величина – это результат некоторого эксперимента, то мат. ожидание - это значение, которое мы будем получать в среднем, если проведём множество (кол-во, стремящееся к бесконечности) независимых экспериментов.
   Обозначение: E(x) – мат. ожидание случайной величины x.
   Обозначение: E(x) – мат. ожидание случайной величины x.
3. *Аддитивность функции*. Говорят, что ф-ия f обладает аддитивностью относительно операции $\oplus$, если для любых a, b, лежащих в области определения f, выполняется:
   $f(a \oplus b) = f(a) \oplus f(b)$
4. *Перекрещивающиеся задачи* – задачи, решающиеся последовательно, причём для решения каждой задачи используются результаты, полученные при решении некоторых или всех предыдущих задач.

   Пример: вычисление чисел Фибоначчи.
   $F_1 = 1, F_2 = 1$
   $F_3 = F_1 + F_2$
   $F_4 = F_2 + F_3$

…
$F_n = F_{n-2} + F_{n-1}$

5. *Динамическое программирование* – метод решения задачи, путём разбиения её на перекрещивающиеся подзадачи.
6. *Стохастическое программирование* – класс методов решения задач в условиях неполной информации (некоторые параметры задачи являются случайными величинами).
7. *Стохастическое динамическое программирование* – стохастический способ решения задачи путём динамического программирования (стохастический и одновременно динамический метод).
8. *GitHub* – популярная платформа для выкладывания кода в интернет в публичном или приватном доступе.

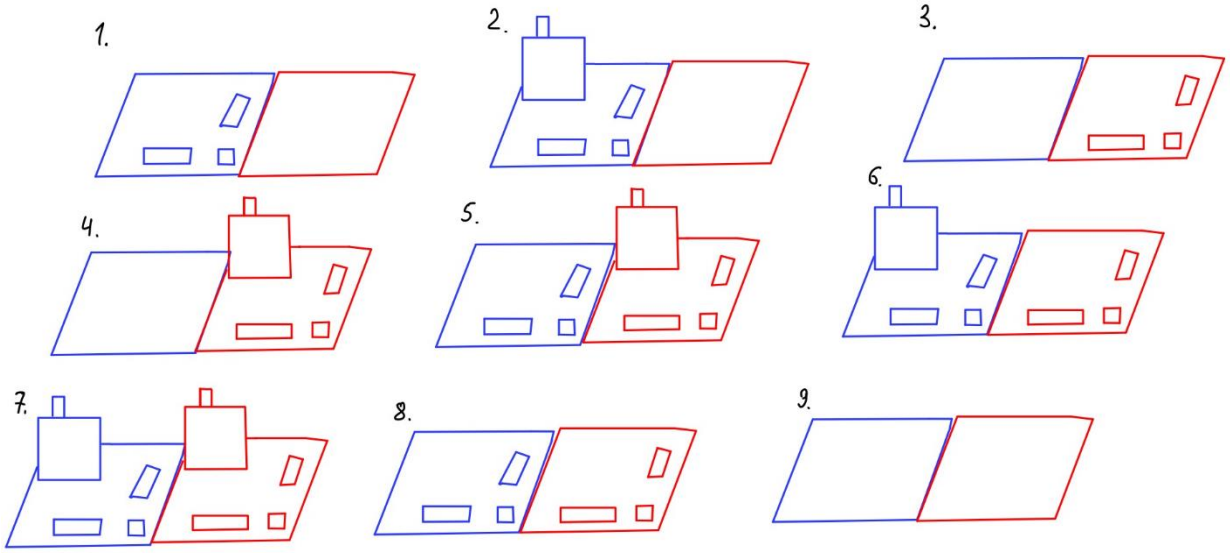
Введение

Стохастическое программирование

Потребность в стохастическом программировании может возникнуть при работе в самых разных областях. Проблема неопределённости параметров часто встречается в экономических, финансовых, научных и инженерных задачах. Чрезвычайно важно обращаться именно к стохастическому программированию при работе с решениями, на которые влияет неопределённость. Примечательно, что большинство проблем принятия решений относятся к этому типу.

Технически стохастические программы гораздо сложнее, чем соответствующие детерминированные аналоги. Следовательно, причины для обращения на практике к стохастическим моделям должны быть очень вескими. Чтобы понять, что такие причины действительно есть, рассмотрим небольшой пример из книги P. Kall и S. Wallace «Stochastic Programming» , показывающий преимущества и недостатки стохастического подхода.

Пусть у бизнесмена есть 2 участка земли. На каждом из них можно построить вспомогательную инфраструктуру. Если на участке построена вспомогательная инфраструктура, то на нём можно построить завод по производству плюшевых мишек. То есть с каждым участком земли можно поступить 3-мя способами: не строить на нём ничего, построить вспомогательную инфраструктуру, построить вспомогательную инфраструктуру и завод. Таким образом, у бизнесмена есть 9 вариантов поведения.



Прибыль с каждого из построенных заводов составит p. Когда строится вспомогательная инфраструктура, p неизвестно. Завод можно построить уже после того, как станет известно p, но тогда его строительство будет стоить дороже на 10%. При этом, чтобы можно было построить завод после того, как станет известно p, на участке уже должна быть вспомогательная

инфраструктура (на то, чтобы построить и инфраструктуру, и завод после объявления p, времени не хватит). В таблице представлены цены строительств инфраструктуры и завода для каждого из участков.

Табл. 1

	Построить инфраструктуру	Построить завод	Построить завод позже
Участок 1	600	200	220
Участок 2	100	600	660

Детерминированное решение

Зачастую такие проблемы решаются следующим образом: определим сначала самое выгодное решение для каждого из возможных значений p (т. е. решим задачу с параметром). Несложно заметить, что:

Табл. 2

p	Номер решения
$p < 700$	9
$700 \leq p < 800$	4
$p > 800$	7

В каждом случае мы считаем, что p определена, поэтому нет смысла переплачивать 10% за строительство завода. Теперь найдём вероятностное распределение, задающее p. Предположим, мы проанализировали имеющиеся данные и пришли к выводу, что p может принимать только 2 значения: 210 и 1250, каждое с вероятностью 0.5. Распространённым методом решения таких задач является замена неопределённой величины на среднее из её возможных значений, взвешенных по вероятности (т. е. на мат. ожидание). Мат. ожидание p равно 730. С помощью таблицы мы видим, что оптимальным будет решение 4 с ожидаемой прибылью: -700 + 730 = 30.

Стохастическое решение

Но действительно ли это решение будет оптимальным? Мы проверим это, найдя математическое ожидание прибыли для каждого возможного варианта поведения. Причём будем действовать максимально эффективно – если на момент объявления p завод ещё не построен, будем принимать решение, строить его или нет в зависимости от того, окупится ли его строительство или лишь увеличит убытки бизнесмена (в реальной жизни бизнесмен действовал бы точно также).

Табл. 3

Решение	Затраты до объявления p	Доход, если p = 210	Доход, если p = 1250	Мат. ожидание прибыли (формула)	Мат. ожидание прибыли (итог)
1	600	0	1250 – 220 = 1030	(0 +1030) / 2 – 600	-85
2	800	210	1250	(210 + 1250) / 2 – 800	-70
3	100	0	1250 – 880 = 590	(0 + 590) / 2 – 100	195
4	700	210	1250	(210 + 1250) / 2 – 700	30
5	1300	210	2500 – 220 = 2280	(210 + 2280) / 2 – 1300	-55
6	900	210	2500 – 660 = 1840	(210 + 1840) / 2 – 900	125
7	1500	420	2500	(420 + 2500) / 2 – 1500	-40
8	700	0	2500 – 220 – 660 = 1620	(0 + 1620) / 2 – 700	110
9	0	0	0	0	0

Мы видим, что математическое ожидание прибыли достигает максимума при решении 3. Решение, которое действительно максимизирует мат. ожидание рассматриваемой величины, называется стохастическим решением.
Примечательно, что ещё 2 решения дают большее мат. ожидание прибыли, чем предложенное изначально. Это показывает преимущество гибких, стохастических решений.
Главная проблема детерминированных решений заключается в том, что любое из них оптимально для какой-то конкретной ситуации, но чаще всего очень плохо для всех остальных. В нашем примере решение 4 оптимально для любого p из отрезка [700, 800). С другой стороны, стохастическое решение редко оказывается лучшим решением, которое можно было бы принять, если бы мы знали, какие значения примут случайные величины. В нашем примере решения «не строить ничего» (с мат. ожиданием прибыли: 0) и «построить сразу оба завода» (с мат. ожиданием прибыли: -40) имеют 50%-ую вероятность

оказаться самыми лучшими. В то время как выбранное нами решение с мат. ожиданием: 195 имеет нулевую вероятность оказаться самым лучшим. Зато такое решение является оптимальным с точки зрения теории вероятностей. Теперь, когда мы показали актуальность стохастического программирования, рассмотрим динамическое программирование.

Динамическое программирование

Задачи, которые можно решить с помощью динамического программирования часто встречаются во всех областях, где используется программирование. Впервые я столкнулся с этой темой ещё на первом курсе, проходя алгоритмы и структуры данных. Классические примеры задач на динамическое программирование – это, например, задача о рюкзаке (её стохастическая версия будет рассмотрена в работе) и задача коммивояжёра. Сформулируем общие принципы динамического программирования для удобства дальнейшей работы.

Определить относительно малую коллекцию подзадач

Показать, как быстро и правильно решать «более крупные» подзадачи с учётом «более мелких».

Показать, как быстро и правильно выводить окончательное решение из решений всех подзадач.

Обычно в качестве подзадач берутся копии исходной задачи только меньшего размера, а дальше строится рекурентное отношение: например, пусть N – некоторый параметр, характеризующий размер задачи (прямо пропорциональный ему), тогда мы определяем, как решить задачу для N = n, если мы знаем решение для N = n – 1 (считая, конечно, что задача меньшего размера – это подзадача большой задачи). Можно заметить, что этот принцип очень похож на математическую индукцию. Действительно, мы осуществляем *переход* от шага N = n – 1 к шагу N = n, чтобы научиться решать задачу для любого N. И как и в математической индукции, нам нужна *база*, от которой будут отталкиваться все последующие шаги. Найдём некоторое n_0, такое что при N < n_0 задача не имеет смыла (обычно это 0 или 1). При N = n_0 задача, если её, конечно, вообще можно решить динамическим программированием становится тривиальной и легко решается. Таким образом, зная решение для N = n_0 и умея из решения для N = n – 1 получать решение для N = n, можем решать задачу для любого N за O(N * k), где k – это время, необходимое для осуществления перехода от n – 1 к n. Также понадобится O(N * m) дополнительной памяти для хранения решений подзадач, где m – оценка памяти, необходимой для хранения решения самой большой подзадачи (для вычисления и хранения базы обычно нужна небольшая константа, которой можно пренебречь). Для лучшей иллюстрации принципов работы динамического программирования в общем случае в начале моего проекта подробно рассмотрен соответствующий пример (задача о взвешенном множестве в путевом графе). Стоит также сказать, что, как уже вскользь замечалось выше, далеко не любая задача может быть решена динамическим

программированием. Исследованию этого вопроса также посвящена одна из рассмотренных в проекте задач (подробнее в Теоретической части на стр. x).

Стохастическое динамическое программирование

Таким образом, проект посвящён применению динамического программирования в условиях стохастической неопределённости.

Требования к программному проекту

Описание функциональных требований

1. Время работы каждой из программ не должно превышать 3 мин.
2. Для каждой решённой задачи нужно представить входные и выходные данные. Программы будут написаны так, чтобы выходные данные выводились в виде связного текста или таблицы. Добиться вывода данных в виде таблицы в Jupiter Notebook можно, например, с помощью библиотеки pandas.
3. Программы должны быть снабжены обильными и подробными комментариями, чтобы упростить понимание принципов их (программ) работы.

Описание нефункциональных требований

1. Для того, чтобы была возможность запустить программы, на компьютере пользователя должен быть установлен Jupiter Notebook версии 5.7.0 и выше, а также Anaconda 3 2020.02 или выше (теоретически всё может сработать и без Anaconda, но гарантии нет). При этом для просмотра любых файлов проекта никакого специального программного обеспечения не требуется (если просмотр осуществляется на платформе GitHub).
2. Пользователь (читатель пособия) должен иметь опыт чтения математической литературы, знать синтаксис язык Python, уметь читать чужой код и иметь хотя бы базовые знания по дискретной математике, теории вероятностей, алгоритмам и структурам данных.

Теоретическая часть

Далее кратко представлена структура проекта, описаны решённые задачи и проанализированные модели.

0. Задача о взвешенном независимом множестве в путевом графе.

Первая рассмотренная в проекте задача – «задача о взвешенном независимом множестве в путевом графе» иллюстрирует принцип работы динамического программирования в общем случае.

Замечание о применимости динамического программирования.

Приведён пример, иллюстрирующий случай, когда динамическое программирование неприменимо. Также было сформулировано достаточное условие того, что задачу можно решать динамическим программированием (все последующие задачи этому условию удовлетворяют): должна выполняться аддитивность функции, сопоставляющей задаче (или подзадаче) её решение, относительно операции композиции подзадач. Говоря математическим языком, пусть f – функция сопоставляющая задаче её решение, $\oplus$ - операция композиции. Тогда для любой задачи x (для которой определено f(x)), представимой в виде композиции конечного числа подзадач $y_1, y_2, \ldots, y_n$ выполняется:

$f(x) = f(y_1 \oplus y_2 \oplus \ldots \oplus y_n) = f(y_1) \oplus f(y_2) \oplus \ldots \oplus f(y_n)$

То есть требуется, чтобы объединение решений подзадач давало корректное решение общей задачи (иначе говоря, предъявляется требование к корректности индукционного перехода).

Модель I. Замена случайной величины математическим ожиданием

Теперь можно переходить к стохастическому динамическому программированию. Идея первой рассматриваемой нами модели крайне проста. Пусть дана некоторая задача, которую теоретически можно решить динамическим программированием, но некоторые параметры заданы в ней не числами, а вероятностным распределением (такая задача называется стохастической). Тогда посчитаем математическое ожидание этих параметров и подставим их в задачу (так делается очень часто, в качестве значения случайной величины берётся её мат. ожидание). То есть мы будем использовать среднее, взвешенное по вероятностям значение этих параметров, что выглядит весьма разумно. Так стохастическая задача превратилась в детерминированную и её можно решить обычным динамическим программированием. Очевидно, что сложность алгоритма никак не изменится ни по времени, ни по памяти.

1. Стохастическая задача о рюкзаке

Затем в работе приведено решение одной из классических задач на динамическое программирование (далее ДП), рассмотренной в стохастическом варианте (когда некоторые параметры являются случайными величинами).

2. ***Азартная игра***
3. ***Задача инвестирования***

В следующих двух разделах решаются избранные задачи из учебника Хэмди А. Тахи «Исследование операций» (гл. 15 Вероятностное динамическое программирование). Задачи эти имеют экономический характер и интересны, в первую очередь, построением математических моделей, к которым можно бы было применить стохастическое динамическое программирование (далее СДП).

4. ***Коварные банковские вклады***

Рассмотрена интересная задача из книги P. Kall и S. Wallace «Stochastic Programming», которая выглядела как самая обычная задача на замену случайных величин мат. ожиданиями, но в итоге потребовала больших интеллектуальных усилий и нестандартного подхода. Не буду вдаваться в подробности (их можно прочесть в самом проекте), скажу лишь, что пришлось реализовать класс для работы с линейным выражением от переменной, которую мы не знаем, но хотим максимизировать. Также в ходе работы над этой задачей проявился один важный недостаток ДП перед классической рекурсией.

Модель II. Максимизация вероятности достижения цели

Пришло время обратиться ко второй модели СДП, рассматриваемой в моей работе - максимизации вероятности достижения цели. До этого мы заменяли вероятностную неопределённость математическим ожиданием (на каждом шаге ДП вместо величины, заданной её вероятностным распределением, мы подставляли её математическое ожидание). Но алгоритмы с использованием мат. ожиданий ориентируются на средние значения, и, по большому счёту, не дают никаких гарантий. Дело в том, что мы не учитываем, насколько и с какой вероятностью реальные значения могут отклоняться от прогнозируемых. Например, пусть X - такая случайная величина, что:

$P(X = 1) = 0.99$,

$P(X = 1000) = 0.01$

Тогда $EX = 0.99 * 1 + 0.01 * 1000 = 10.99$, и именно на это значение мы будем "рассчитывать" в программах, написанных по первой модели. Но при этом в 99% случаев X = 1. Таким образом, есть потребность в методе, дающем большие гарантии.

Все рассматриваемые нами задачи сводятся к максимизации (или минимизации) значения некоторой переменной. Пусть требуется максимизировать X. Выберем X_{inf} - наименьшее удовлетворяющее нас значение X и будем стараться максимизировать $P(X \geq X_{inf})$ - вероятность того, что X не меньше, чем X_{inf}. В особых случаях (в задачах с повышенными требованиями к "надёжности" решения) можно даже потребовать, чтобы $P(X \geq X_{inf})$ была не меньше некоторого значения P_{wish}, и тогда алгоритм должен будет искать стратегию для достижения $P(X \geq X_{inf}) \geq P_{wish}$. Аналогично в задаче минимизации будем максимизировать $P(X \leq X_{sup})$, где X_{sup} - наибольшее

удовлетворяющее нас значение X. Эта модель и называется *максимизация вероятности достижения цели*.

5. Инвестирование продолжается

Задача инвестирования, взятая из учебника Хэмди А. Тахи «Исследование операций» (гл. 15 Вероятностное динамическое программирование) решается с использованием алгоритма, основанного на модели максимизации достижения прибыли. Сама задача несложная, но она иллюстрирует работу весьма сложной модели на практике.

Практическая часть

Проект выполнен в среде разработки Jupiter Notebook – инструменте для создание аналитических отчётов, а также написания лабораторных, исследовательских или проектных работ в области компьютерных наук, обладающем широким набором возможностей для визуализации данных. Данная платформа была выбрана для работы над проектом, поскольку в одном файле формата ipynb могут содержаться и фрагменты кода, и вывод результата их работы, и текстовые заметки. Код пишется на языке Python, пакет Anaconda представляет большое количество специальных библиотек, облегчающих работу с данными и сложными математическими операциями. Для создания текстовых заметок в Notebook используется язык разметки Markdown, основанный на системе компьютерной вёрстки TeX, и все основные команды аналогичны таковым в LaTeX. Таким образом, Markdown является удобным инструментом для набора текста, содержащего большое количество математических символов. Также Jupiter Notebook предоставляет возможность набирать текст непосредственно на html. Иногда удобно писать, делая вставки html в Markdown – это расширяет спектр возможностей, хотя и понижает читаемость исходного кода заметки. После завершения работы с файлом, он может быть переведён в формат pdf. Эта система представляется очень удобной для формата работы, когда решаются не связанные друг с другом и не очень большие по объёму задачи. В проекте каждую задачу предваряет её условие, обсуждение решения, построение математической модели, если нужно. После каждой задачи будут представлены результаты работы программы и какие-то выводы или замечания, если это релевантно.

Таким образом, файл с задачами представляет собой документ формата ipynb, содержащий объяснения принципов работы, математические модели, постановку задач, предложения и доказательства гипотез, непосредственно сам код запрограммированных задач, входные и выходные данные, а также некоторые выводы.

В репозитории проекта будут находиться 4 файла: README с краткой пояснительной информацией, данный отчёт, Tasks.ipynb – файл с задачами и мат. моделями, который можно открыть в Jupiter Notebook, чтобы запустить код, и Tasks.pdf – этот же файл, конвертированный в pdf для удобства просмотра.

Перспективы дальнейшей работы

Проект имеет обширные перспективы дальнейшей работы. Наиболее интересным представляется продолжение исследования модели максимизации достижения цели, поиск более оптимальных алгоритмов. Приведённая в проекте задача на эту тему решена довольно наивным и неэффективным алгоритмом с экспоненциальной асимптотикой по времени, и существуют различные приёмы, которые можно попробовать применить для ускорения времени работы. Например, можно попробовать идти от конца в начало и запоминать результат для каждого этапа (похожий приём помог нам добиться хороших результатов в самой первой задаче проекта). Но мы столкнёмся с той же проблемой, с которой столкнулись в задаче о «коварных банковских вкладах» - идя с начала, мы не знаем, с какими числами мы работаем, и приходиться прибегать к различным сложно реализуемым ухищрениям (можно попробовать работать не с конкретными числами, а с классами, содержащими линейное выражение от некоторой переменной (см. раздел «4. Коварные банковские вклады» в самом проекте)). В общем, тема сложная и интересная, и много чего ещё может быть рассмотрено и реализовано.

Также в качестве продолжения проекта было бы очень интересно рассмотреть какую-то большую задачу из реальной жизни, и посмотреть, чего в ней можно добиться с помощью стохастического динамического программирования. Мой научный руководитель, Хамисов О. В., обещал поискать для меня такую задачу, если мы решим продолжать работу над проектом.

Список литературы

1. Хэмди А. Таха «Исследование операций»
2. P. Kall and S. Wallace «Stochastic Programming»
3. В. Д. Власенко «Динамическое и стохастическое программирование»
4. В. Е. Гмурман «Теория вероятностей и математическая статистика»