Теория алгоритмов

Практика

Владимир Анатольевич Судаков

2026 г.

Правила

- Решение домашнего задания (ДЗ) нужно приносить на следующее занятие. Если решение – это программа, то присылайте ссылку на github.
- Лучше не полное решение ДЗ, чем его отсутствие.
- Можно и нужно спрашивать по почте и на занятии. Лучше если вопрос будет коротким.
- У нас будут контрольные, но когда – заранее неизвестно
- Опоздание больше 15 минут – пропуск.
- За активность/ответы на занятиях будут дополнительные плюсы.
- Вовремя решенные ДЗ, контрольные без ошибок, и не более 2-х пропусков – автомат на экзамене

Давайте познакомимся

- Дано:
  - список группы ФИО через пробел по строкам
- Требуется найти повторения имен:
  - вывести по строкам ФИО<пробел><число повторений имени>
- Какая группа «хорошая»?
- Давайте сравним решения (и Ваши и LLM):
  - по каким метрикам будем сравнивать?
- Какая LLM хорошая?

Задача о паросочетаниях (stable matching problem)

Подбор больниц для студентов медицинских вузов

- Цель. Имея набор предпочтений среди больниц и студентов медицинских вузов, разработать процесс приема.
- Неустойчивая пара. Больница h и студент s образуют неустойчивую пару, если оба: h отдает предпочтение s перед одним из принятых студентов. s отдает предпочтение h перед назначенной больницей.
- Стабильное распределение. Распределение без нестабильных пар. Естественное и желательное состояние.

Дано

- Входные данные: множество из n больниц H и множество из n студентов S (пока считаем, что больница берет одного только студента). Каждая больница h ранжирует студентов. Каждый студент s ранжирует больницы.

favorite ↓ least favorite ↓

	1st	2nd	3rd
Atlanta	Xavier	Yolanda	Zeus
Boston	Yolanda	Xavier	Zeus
Chicago	Xavier	Yolanda	Zeus

hospitals' preference lists

favorite ↓ least favorite ↓

	1st	2nd	3rd
Xavier	Boston	Atlanta	Chicago
Yolanda	Atlanta	Boston	Chicago
Zeus	Atlanta	Boston	Chicago

students' preference lists

Идеальное паросочетание

	1st	2nd	3rd
Atlanta	Xavier	Yolanda	Zeus
Boston	Yolanda	Xavier	Zeus
Chicago	Xavier	Yolanda	Zeus

	1st	2nd	3rd
Xavier	Boston	Atlanta	Chicago
Yolanda	Atlanta	Boston	Chicago
Zeus	Atlanta	Boston	Chicago

Нестабильные пары

	1st	2nd	3rd
Atlanta	Xavier	Yolanda	Zeus
Boston	Yolanda	Xavier	Zeus
Chicago	Xavier	Yolanda	Zeus

	1st	2nd	3rd
Xavier	Boston	Atlanta	Chicago
Yolanda	Atlanta	Boston	Chicago
Zeus	Atlanta	Boston	Chicago

A–Y is an unstable pair for matching M = { A–Z, B–Y, C–X }

Тест 1

- Какая пара является неустойчивой в паросочетании { A–X, B–Z, C–Y }?
- **A.** A–Y.
- **B.** B–X.
- **C.** B–Z.
- **D.** Ни один из перечисленных вариантов.

	1st	2nd	3rd
Atlanta	Xavier	Yolanda	Zeus
Boston	Yolanda	Xavier	Zeus
Chicago	Xavier	Yolanda	Zeus

	1st	2nd	3rd
Xavier	Boston	Atlanta	Chicago
Yolanda	Atlanta	Boston	Chicago
Zeus	Atlanta	Boston	Chicago

Домашнее задание

- Напишите программу которая определяет является ли паросочетание устойчивым
- Вход – таблицы предпочтений и паросочетания.
- Выход – ответ: Устойчивое или Неустойчивое.

Упражнение 1

- Имеется город, в котором n мужчин и n женщин пытаются вступить в брак. У каждого мужчины имеется список предпочтений, содержащий оценки всех женщин, а у каждой женщины — список предпочтений с оценками всех мужчин.
- Множество всех $2n$ людей разделено на две категории: хорошие люди и плохие люди. Предположим, для некоторого числа k ($1 \leq k \leq n - 1$) имеются k хороших мужчин и k хороших женщин; следовательно, существует n-k плохих мужчин и n-k плохих женщин.
- Каждый участник предпочитает любого хорошего человека любому плохому человеку. Формально каждый список предпочтений обладает тем свойством, что оценка любого хорошего человека противоположного пола выше оценки любого плохого человека противоположного пола: первые к позиций в нем занимают хорошие люди (противоположного пола) в некотором порядке, а затем идут n-k плохих людей (противоположного пола) в некотором порядке.
- Покажите, что в каждом устойчивом паросочетании каждый хороший мужчина вступает в брак с хорошей женщиной.

Упражнение 2(1)

Решите, является ли приведенное ниже утверждение истинным или ложным. Если оно истинно, приведите короткое объяснение; если ложно — приведите контрпример:

1. В любой конфигурации задачи устойчивых паросочетаний существует устойчивое паросочетание, содержащее пару (m, w), в котором m находится на первом месте в списке предпочтений w, а w находится на первом месте в списке предпочтений m.

Упражнение 2(2)

Решите, является ли приведенное ниже утверждение истинным или ложным. Если оно истинно, приведите короткое объяснение; если ложно — приведите контрпример:

2. Допустим, имеется конфигурация задачи устойчивых паросочетаний с мужчиной m и женщиной w, для которых m находится на первом месте в списке предпочтений w, а w находится на первом месте в списке предпочтений m. В этом случае в каждом устойчивом паросочетании S для этой конфигурации пара (m, w) принадлежит S.

Упражнение 3

Существует много ситуаций, в которых можно задавать вопросы, относящиеся к некой разновидности принципа «устойчивости». В следующем примере используется формулировка, основанная на конкуренции между двумя компаниями.

Допустим, имеются две телевизионные сети; назовем их A и B. Также доступны n программных окон в течение времени массового просмотра, а каждая сеть проводит n телевизионных передач. Каждая сеть хочет создать расписание — схему распределения передач по разным окнам, чтобы обеспечить себе как можно большую долю рынка.

Следующий способ поможет сравнить, насколько хорошо две сети справляются со своей задачей, исходя из их расписания. У каждой передачи имеется фиксированный рейтинг, основанный на количестве людей, посмотревших передачу в прошлом году; будем считать, что рейтинги двух передач никогда не совпадают. Сеть выигрывает заданное окно, если передача, которую она ставит в это окно, обладает более высоким рейтингом, чем передача в расписании других сетей для этого окна. Цель каждой сети — захватить как можно больше окон.

Допустим, в первую неделю осеннего сезона сеть A публикует расписание S, а сеть B — расписание T. На основе этой пары расписаний каждая сеть захватывает некоторые окна в соответствии с приведенным выше правилом. Пара расписаний (S, T) будет называться устойчивой, если ни одна из сетей не может в одностороннем порядке изменить свое расписание и получить дополнительные окна. Иначе говоря, не существует расписания S' такого, что сеть A с парой (S', T) получит больше окон, чем с парой (S, T). По соображениям симметрии также не существует расписания T', с которым сеть B со своей парой (S, T') захватит больше окон, чем с парой (S, T).

Аналогия вопроса Гейла и Шепли для подобной устойчивости выглядит так: можно ли утверждать, что для каждого множества телепередач и оценок всегда существует устойчивая пара расписаний? Чтобы ответить на этот вопрос, сделайте одно из двух:

(а) приведите алгоритм, который для любого множества передач и связанных с ними оценок строит устойчивую пару расписаний;

или

(б) приведите пример множества передач и связанных с ними оценок, для которого не существует устойчивой пары расписаний.

Упражнение 4

Гейл и Шепли опубликовали свою статью о задаче устойчивых паросочетаний в 1962 году; однако разновидность их алгоритма к тому времени уже почти 10 лет использовалась Национальной программой распределения студентов- медиков по больницам. По сути, ситуация выглядела так: было т больниц, в каждой из которых имелось некоторое количество вакансий. Также было п студентов-медиков, которые получали диплом и были заинтересованы в поступлении в одну из больниц. Каждая больница строила оценки студентов в порядке предпочтения, и каждый студент строил оценки больниц в порядке предпочтения. Будем считать, что количество выпускников превышает количество доступных вакансий в *m* больницах. Естественно, задача заключалась в поиске способа связывания каждого студента не более чем с одной больницей, при котором были бы заполнены все вакансии во всех больницах. (Так как предполагается, что количество студентов превышает количество вакансий, некоторые студенты не будут приняты ни в одну из больниц.)

Распределение студентов по больницам называется устойчивым, если не встречается ни одна из следующих ситуаций:

1. Неустойчивость первого типа: имеются студенты s и s', а также больница h, для которых:

- s назначается в h,
- s' не назначается ни в одну больницу,
- h предпочитает s' студенту s.

2. Неустойчивость второго типа: имеются студенты s и s', а также больницы h и h', для которых:

- s назначается в h,
- s' назначается в h',
- h предпочитает s' студенту s,
- s предпочитает h больнице h'.

Фактически ситуация повторяет задачу устойчивых паросочетания, если не считать того, что 1) в больницах обычно открыто более одной вакансии и 2) студентов больше, чем вакансий.

Покажите, что всегда существует устойчивое распределение студентов по больницам, и приведите алгоритм поиска такого распределения.

Упражнение 5

Могут ли мужчина или женщина улучшить свое положение, солгав относительно своих предпочтений?

Допустим, что у каждого участника имеется истинный порядок предпочтений. Теперь рассмотрим женщину w. Предположим, w предпочитает мужчину m мужчине m', но и m и m' находятся на нижних позициях ее списка предпочтений. Может ли случиться так, что переключение порядка m и m' в списке предпочтений (например, ложным утверждением о том, что она предпочитает m' мужчине m) и выполнением алгоритма с ложным списком предпочтений w окажется в паре с мужчиной m'', которого она в действительности предпочитает как m, так и m' ?

Тот же вопрос можно задать и для мужчин, но в контексте нашего упражнения мы ограничимся женщинами.

Чтобы ответить на этот вопрос, сделайте одно из двух:

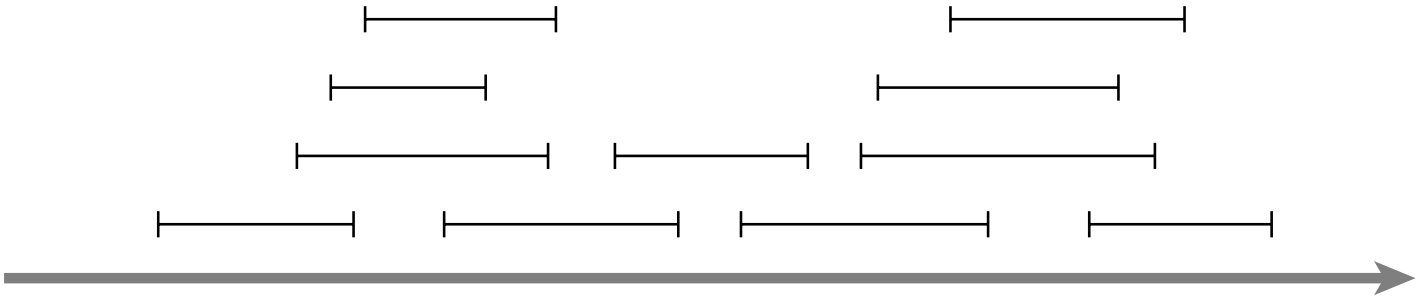
(а) приведите доказательство того, что для любого множества списков предпочтений изменение порядка следования пары в списке предпочтений не сможет улучшить статус партнера женщины в алгоритме Гейла-Шепли;

или

(б) приведите пример множества списков предпочтений, для которых существует изменение порядка следования пары, способное улучшить статус партнера женщины.

Интервальное планирование

Имеется некий ресурс — аудитория, суперкомпьютер, электронный микроскоп и т. д.; множество людей обращается с заявками на использование ресурса в течение периода времени. Заявка имеет вид: «Можно ли зарезервировать ресурс, начиная с времени s и до времени f?» Будем считать, что ресурс в любой момент времени может использоваться только одним человеком. Планировщик выбирает подмножество заявок, отклоняя все остальные, чтобы принятые заявки не перекрывались по времени. Целью планирования является максимизация количества принятых заявок.

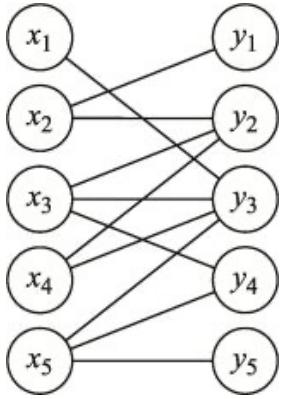


Взвешенное интервальное планирование

Интервалу каждой заявки i присваивается соответствующий коэффициент *(вес)* $v_i > 0$; можно рассматривать его как оплату, которую владелец ресурса получит от i-го претендента при удовлетворении его заявки. Наша цель — найти совместимое подмножество интервалов с максимальным общим весом.

Двудольные паросочетания

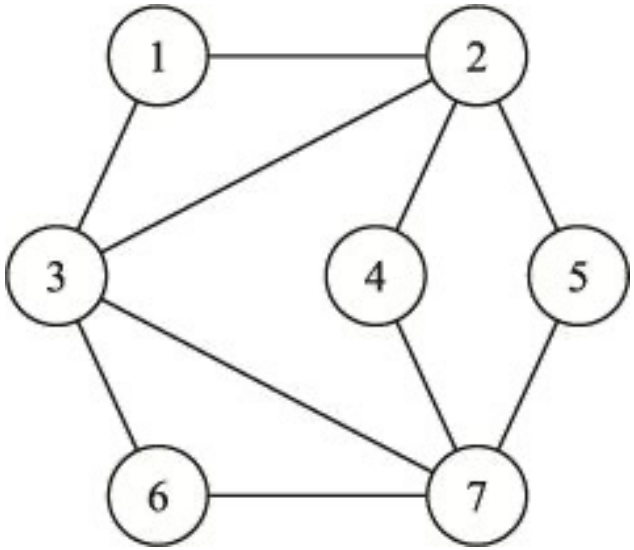
Граф G = (V,E) называется двудольным (bipartite), если множество узлов V может быть разбито на множествами таким способом, что у каждого ребра один конец принадлежит X, а другой конец — Y.



В задаче поиска устойчивых паросочетаний результат строился из пар мужчин и женщин. В случае двудольных графов ребра представляются парами узлов, поэтому мы можем сказать, что паросочетание в графе G = (V,E) представляет собой множество M ребер E с тем свойством, что каждый узел входит не более чем в одно ребро. Паросочетание M является идеальным, если каждый узел входит ровно в одно ребро.

Независимое множество

- Для заданного графа *G = (V,E)* множество узлов *S* из *V* называется ***независимым***, если никакие два узла, входящие в *S*, не соединяются ребром.
- Задача поиска независимого множества формулируется так: для заданного графа *G* найти независимое множество максимально возможного размера.



Сведение задачи

- Задачи интервального планирования и паросочетаний в двудольном графе могут рассматриваться как особые случаи задачи независимого множества.
- Для задачи интервального планирования определите граф G = (F, *E),* в котором узлы соответствуют интервалам, а каждая перекрывающаяся пара соединяется ребром; независимые множества в *G* соответствуют совместимым подмножествам интервалов.
- Для задачи паросочетаний в двудольном графе:
  - Для заданного двудольного графа *G'=(V', E')* выбираемые объекты соответствуют ребрам, а конфликты возникают между двумя ребрами, имеющими общий конец (такие пары ребер не могут принадлежать общему паросочетанию).
  - Соответственно мы определяем граф G = (V, *E),* в котором множество узлов *V* эквивалентно множеству ребер *E' в G'.* Мы определяем ребро между каждой парой элементов *V*, соответствующих ребрам G' с общим концом.
  - Теперь можно проверить, что независимые множества *G* точно соответствуют паросочетаниям G'.
  - Посторим?

Поиск независимого множества

Эффективные алгоритмы для ее решения неизвестны, и есть гипотеза, что такого алгоритма не существует.

- Очевидный алгоритм, действующий методом «грубой силы», перебирает все подмножества узлов, проверяет каждое из них на независимость, после чего запоминает самое большое из обнаруженных подмножеств.
- Возможно, это решение близко к лучшему из того, что возможно сделать в этой задаче.

Задача независимого множества принадлежит к большому классу задач, называемых *NP-полными.*

- Ни для одной из этих задач не известен эффективный алгоритм решения, но все они эквивалентны в том смысле, что решение любой такой задачи будет точно подразумевать существование такого решения у всех.

Можно ли сказать хоть что-нибудь положительное о сложности задачи независимого множества?

- Если имеется граф G из 1000 узлов и мы захотим убедить вас в том, что он содержит независимое множество S с размером 100, сделать это будет несложно. Достаточно показать граф G, выделить узлы S красным цветом и позволить вам убедиться в том, что никакие два узла не соединены ребром.
- Между проверкой того, что множество действительно является большим независимым, и непосредственным поиском большого независимого множества существует огромная разница.

В некоторых особо сложных задачах даже может не быть простого способа «проверки» решений в этом смысле

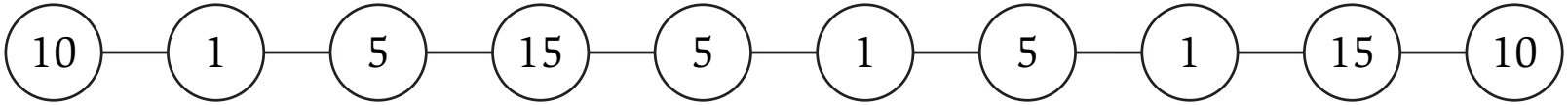
Задача конкурентного размещения

- Допустим, имеются две крупные компании, которые открывают свои сети кофеен по всей стране (назовем их JavaPlanet и Queequeg's Coffee); в настоящее время они соревнуются за свою долю рынка в некотором географическом регионе. Сначала JavaPlanet открывает свое заведение; затем Queequeg’s Coffee открывает свое; потом JavaPlanet; снова Queequeg’s, и т. д. Допустим, они должны выполнять градостроительные нормы, согласно которым две кофейни не должны располагаться слишком близко друг к другу, и каждая компания пытается разместить свои заведения в самых удобных точках. Кто выиграет?

- Сделаем правила «игры» более конкретными. Географический регион, о котором идет речь, разделен на *n* зон с номерами 1, 2,..., n. Каждой зоне i присваивается оценка b_j — доход, который получит любая из компаний, открывшая кофейню в этой зоне. Наконец, некоторые пары зон (i,j) являются смежными, а местные градостроительные нормы запрещают открывать кофейни в двух смежных зонах независимо от того, какой компании они принадлежат (а также запрещают открывать две кофейни в одной зоне). Для моделирования этих конфликтов мы воспользуемся графом G = (F, *E*), где *V* — множество зон, а (i,j) — ребро в *E*, если зоны i и *j* являются смежными. Таким образом, градостроительные нормы требуют, чтобы полный набор кофеен образовал независимое множество в *G.*

- В нашей игре два игрока $\boldsymbol{P_1}$ и $\boldsymbol{P_2}$ поочередно выбирают узлы в G, игрок $\boldsymbol{P_1}$ делает первый ход. Набор всех выбранных узлов в любой момент времени должен образовать независимое множество в G. Предположим, игрок $\boldsymbol{P_2}$ имеет целевую границу *B*, и мы хотим узнать: существует ли для $\boldsymbol{P_2}$ такая стратегия, чтобы независимо от выбора ходов $\boldsymbol{P_1}$ игрок $\boldsymbol{P_2}$ мог выбрать множество узлов с суммарной оценкой не ниже *B*?

Пример конкурентного размещения



Предположим, игроку $\boldsymbol{P_2}$ установлена целевая граница $\boldsymbol{B}$ = 20. В этом случае у $\boldsymbol{P_2}$ имеется выигрышная стратегия. С другой стороны, при $\boldsymbol{B}$ = 25 у игрока $\boldsymbol{P_2}$ такой стратегии нет.

Класс задачи конкурентного размещения

- Задача конкурентного размещения отличается от задачи независимого множества, в которой, насколько нам известно, найти большое решение сложно, зато предложенное большое решение проверяется легко.
- Эти отличия формализуются в классе PSPACE-полых задач, к которому принадлежит задача конкурентного размещения.
- Считается, что PSPACE-полные задачи строго сложнее NP-полных задач, и это предполагаемое отсутствие коротких «доказательств» их решений является признаком этой повышенной сложности.
- Концепция PSPACE-полноты отражает широкий спектр задач, связанных с играми и планированием; многие из них считаются фундаментальными проблемами в области искусственного интеллекта.

Домашнее задание 3

Одна из задач на выбор:

1. Независимое множество
2. Конкурентное размещение
3. Двудольные паросочетания
4. Взвешенное интервальное планирование

Написать программу решающую задачу.

Оценить время решения.

Домашнее задание 3

Как оценить время:

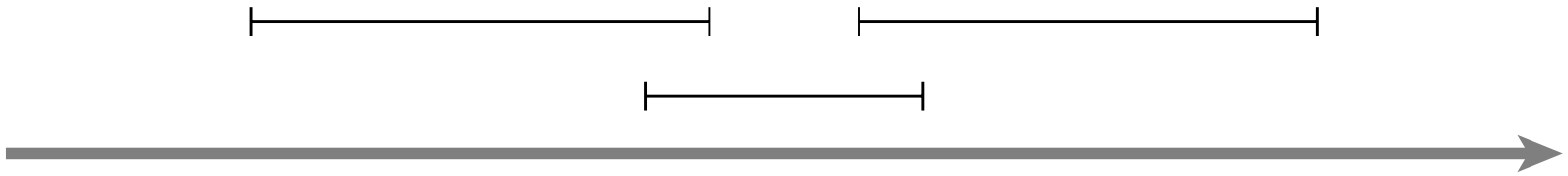
- Эмпирически (на базе экспериментов). Оценить математическое ожидание и СКО.
- Построить график зависимости времени решения от размера входных данных *n* (оценка среднего – линия, 2 СКО – трубка). Должен быть виден характер зависимости.
- Построить гистограмму времени решения для выбранного *n* (подумайте как его выбрать).
- Выбрать гипотезу о законе распределения времени для выбранного *n* и проверить ее.
- Ответить на вопрос: является ли полученное время «хорошим»? Обосновать свой ответ.

Решение задачи интервального планирования

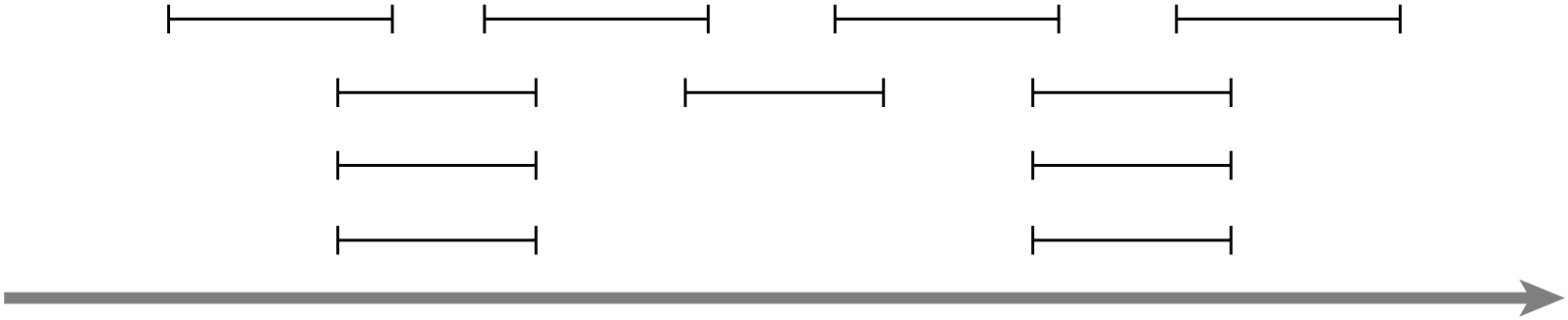
- жадный алгоритм
  - Что будем выбирать жадно?

Решение (1)

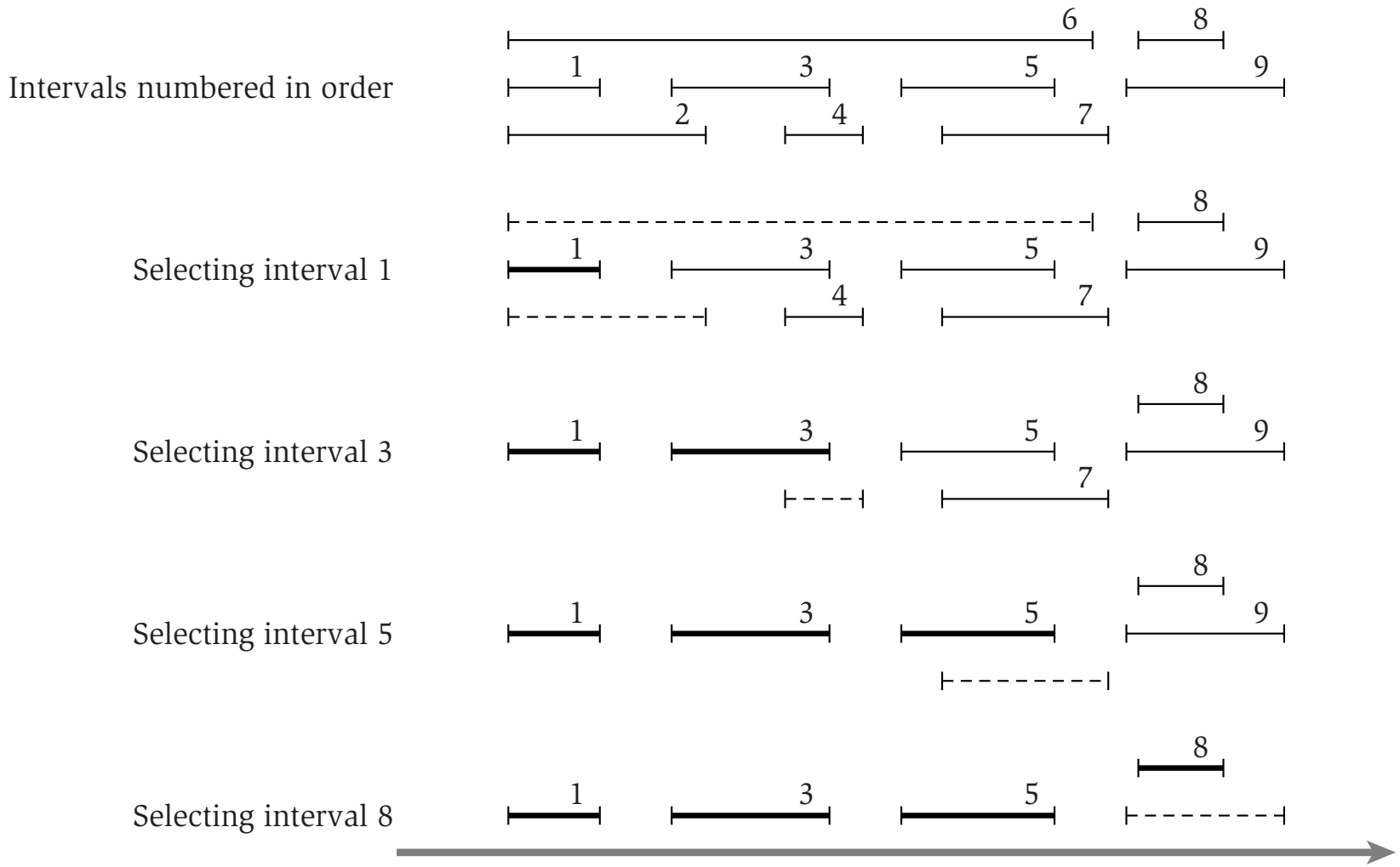
Решение (2)



Решение (3)



Решение (4)


Intervals numbered in order
Selecting interval 1
Selecting interval 3
Selecting interval 5
Selecting interval 8
1
2
3
4
5
6
7
8
9

Упражнение 6

Давайте посмотрим на https://github.com/sudakov/algo-lab/blob/main/gale_shapley.ipynb

- Можно ли обойтись без women_rankings?
- Какая временная сложность у алгоритма?
- В чем недостаток данной программной реализации? И как его устранить?

Упражнение 7

Расположите функции из следующего списка по возрастанию скорости роста. Другими словами, если функция $g(n)$ в вашем списке следует непосредственно после $f(n)$, из этого следует, что $f(n) = O(g(n))$.

$$f_1(n) = 10^n$$

$$f_2(n) = n^{1/3}$$

$$f_3(n) = n^n$$

$$f_4(n) = \log_2 n$$

$$f_5(n) = 2^{\sqrt{\log_2 n}}$$

Упражнение 8

Пусть f и g — две функции, получающие неотрицательные значения, и $f = O(g)$. Покажите, что $g = \Omega(f)$

Упражнение 9

В приведенном ниже списке указано время выполнения пяти алгоритмов (предполагается, что указано *точное* время выполнения). Насколько медленнее будет работать каждый из алгоритмов, если:

(а) увеличить размер входных данных вдвое;

(б) увеличить размер входных данных на 1?

(a) n^2

(b) n^3

(c) $100n^2$

(d) $n \log n$

(e) 2^n

Упражнение 10

Есть шесть алгоритмов, время выполнения которых приведено ниже. Предполагается, что указано точное количество выполняемых операций как функция размера данных *n.* Предположим, имеется компьютер, способный выполнять 10^{10} операций в секунду, и результат должен быть вычислен не более чем за час. При каком наибольшем размере входных данных *n* результат работы каждого алгоритма может быть вычислен за час?

(a) n^2

(b) n^3

(c) $100n^2$

(d) $n \log n$

(e) 2^n

(f) 2^{2^n}

Упражнение 11

Расположите функции из следующего списка по возрастанию скорости роста. Другими словами, если функция $g(n)$ в вашем списке следует непосредственно после $f(n)$, из этого следует, что $f(n) = O(g(n))$.

$$f_1(n) = n^{2.5}$$

$$f_2(n) = \sqrt{2n}$$

$$f_3(n) = n + 10$$

$$f_4(n) = 10^n$$

$$f_5(n) = 100^n$$

$$f_6(n) = n^2 \log n$$

Упражнение 12

Расположите функции из следующего списка по возрастанию скорости роста. Другими словами, если функция $g(n)$ в вашем списке следует непосредственно после $f(n)$, из этого следует, что $f(n) = O(g(n))$.

$$g_1(n) = 2^{\sqrt{\log n}}$$

$$g_2(n) = 2^n$$

$$g_4(n) = n^{4/3}$$

$$g_3(n) = n(\log n)^3$$

$$g_5(n) = n^{\log n}$$

$$g_6(n) = 2^{2^n}$$

$$g_7(n) = 2^{n^2}$$

Упражнение 13

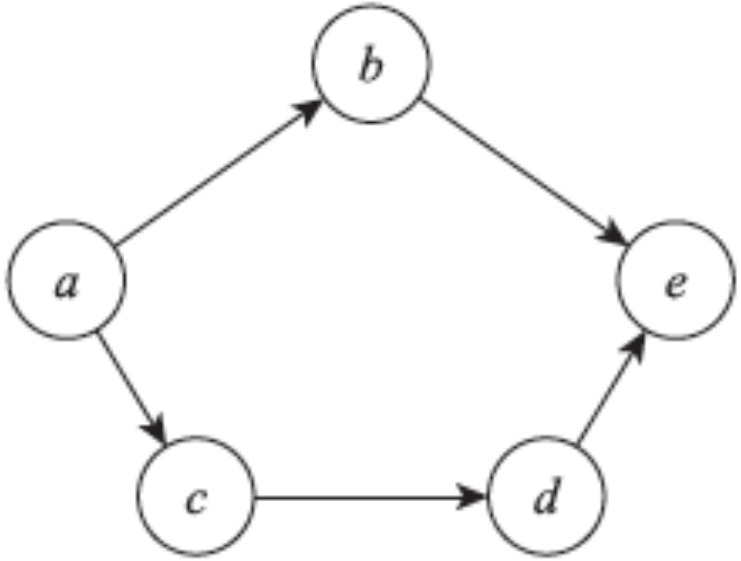
- Предложите алгоритм для проверки наличия циклов в заданном ненаправленном графе. Если граф содержит цикл, то ваш алгоритм должен вывести его.
- Выводить все циклы в графе не нужно, достаточно одного.
- Алгоритм должен выполняться за время $O(m + n)$ для графа с n узлами и m ребрами.

Упражнение 14

- Вдохновившись примером великого Владимира Набокова, Ваши друзья занялись лепидоптерологией (изучением бабочек). Часто при возвращении из поездки с новыми образцами им бывает очень трудно определить, сколько же разных видов было поймано, ведь многие виды очень похожи друг на друга.
- Вашим друзьям хотелось бы разделить n образцов на две группы (относящиеся к видам A и B), но напрямую классифицировать каждый отдельно взятый образец очень трудно. Вместо этого они решают действовать немного иначе.
- Каждая пара образцов i и j располагается рядом и тщательно изучается. Если исследователи достаточно уверены в своей оценке, они помечают образцы из пары (i, j) либо как «одинаковые» (то есть как относящиеся к одному виду), либо как «разные» (то есть как относящиеся к разным видам). Также они могут отказаться от назначения оценки конкретной паре; в таком случае пара называется неопределенной. Итак, имеется набор из n образцов, а также набор из m оценок («одинаковые» или «разные») для пар, которые не были отнесены к неопределенным.
- Вашим друзьям хотелось бы знать, соответствуют ли эти данные представлению о том, что каждая бабочка относится к одному из видов A или B. Говоря конкретнее, набор из m оценок объявляется непротиворечивым, если возможно пометить каждый образец A или B так, чтобы для каждой пары (i, j) с пометкой «одинаковые» i и j относились к одному виду, а для каждой пары (i, j) с пометкой «разные» i и j относились к разным видам. Ваши друзья возятся с проверкой непротиворечивости оценок, как вдруг один из них осознает: ведь вы сможете придумать алгоритм, который позволит с ходу получить ответ на этот вопрос.
- Предложите алгоритм для проверки непротиворечивости оценок с временем выполнения $O(m + n)$.

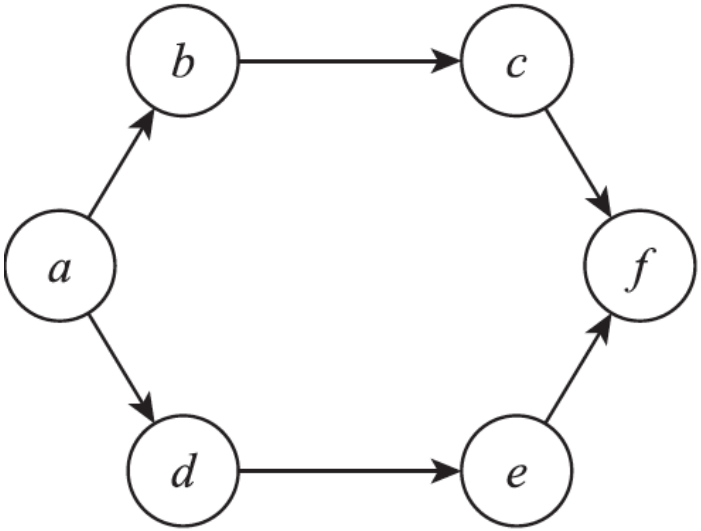
Упражнение 15

Сколько топологических упорядочений существует у этого графа?



Упражнение 16

Рассмотрим направленный ациклический граф G



Сколько топологических упорядочений он имеет?

Упражнение 17

- Ваши друзья открывают компанию, занимающуюся компьютерной безопасностью, и им необходимо получить лицензии на n разных криптографических продуктов. Согласно действующим требованиям, они могут получать не более одной лицензии в месяц.
- Каждая лицензия в настоящее время стоит \$100, но со временем цена растет по экспоненциальному закону: в частности, лицензия j ежемесячно умножается на коэффициент $r_j > 1$ (где r_j — заданный параметр). Таким образом, если лицензия j приобретается через t месяцев, она будет стоить $100 \cdot r_j{}^t$. Будем считать, что все скорости роста цены различны, то есть $r_i \neq r_j$ для $i \neq j$ (хотя все они начинаются с одной исходной цены \$100).
- Вопрос: если компания может покупать не более одной лицензии в месяц, в каком порядке следует совершать покупки, чтобы общая потраченная сумма была минимальной?
- Приведите алгоритм, который получает n скоростей роста цены $r_1, r_2, ..., r_n$ и вычисляет порядок покупки лицензий, чтобы общая сумма потраченных денег была минимальной. Время выполнения алгоритма должно быть полиномиальным по n.

Упражнение 17(2)

Предположим, вместо покупки лицензий с возрастающей ценой вы пытаетесь продавать оборудование, цена которого последовательно снижается.

Цена элемента i снижается с коэффициентом $r_i < 1$ в месяц, начиная с \$100, так что при продаже его через t месяцев Вы получите $100 \cdot r_j{}^t$.

Иначе говоря, скорость роста теперь не больше, а меньше 1.

Если в месяц продается только один вид оборудования, в каком оптимальном порядке их следует продавать?

Упражнение 18

Ваши друзья занялись расширяющейся областью анализа временных рядов. Целью анализа является поиск закономерностей в последовательностях событий, происходящих во времени. Одним из источников таких данных с естественным упорядочением по времени могут служить покупки на биржах (что и когда покупается). Ваши друзья хотят иметь эффективный механизм обнаружения некоторых «закономерностей» в заданной последовательности S таких событий: например, они хотят знать, происходят ли в последовательности S четыре события

```
купить Yahoo, купить eBay, купить Yahoo, купить Oracle
```

в такой очередности (но не обязательно подряд).

Они начинают с набора всех возможных событий (например, возможных транзакций) и последовательности S из n таких событий. Заданное событие может встречаться в S несколько раз (например, акции Yahoo могут покупаться многократно в одной последовательности S). Последовательность S' называется подпоследовательностью S, если из S возможно удалить некоторые события так, чтобы оставшиеся события (с соблюдением порядка) были равны последовательности S'. Так, приведенная последовательность из четырех событий является подпоследовательностью для последовательности

```
купить Amazon, купить Yahoo, купить eBay, купить Yahoo, купить Yahoo, купить Oracle
```

Ваши друзья хотят вводить короткие последовательности и быстро узнавать, являются ли они подпоследовательностями S. Поставлена следующая задача: предложите алгоритм, который получает две последовательности событий — S' длины m и S длины n (в обеих последовательностях события могут повторяться) и за время O(m + n) решает, является ли S' подпоследовательностью S.

Упражнение 19

Представьте длинную, спокойную проселочную дорогу, на которой редко встречаются отдельные дома. Будем считать, что дорога представляет собой отрезок прямой с двумя концами, восточным и западным. Также предположим, что несмотря на пасторальную обстановку, жители этих домов активно пользуются мобильными телефонами. Требуется разместить базовые станции сотовой связи вдоль дороги так, чтобы любой дом находился на расстоянии не более четырех километров от одной из базовых станций.

Предложите эффективный алгоритм для достижения этой цели с минимальным количеством базовых станций.

Упражнение 20

Ваш друг работает в детском лагере и отвечает за проведение мероприятий в группе подростков. В его планы входит проведение мини-триатлона: каждый участник должен проплыть 20 дорожек в бассейне, проехать на велосипеде 10 км., а потом пробежать 3 км. Участники будут выходить на дистанцию поочередно, по следующему правилу: в любой момент времени бассейн будет использоваться только одним участником. Другими словами, первый участник проплывает 20 дорожек, выходит и садится на велосипед. Как только он покидает бассейн, второй участник начинает проплывать свои 20 дорожек; когда он выходит из бассейна, заплыв начинает третий участник… и т. д.

У каждого участника имеется прогнозируемое время плавания (время, за кото-рое он должен проплыть 20 дорожек), прогнозируемое время проезда (время, за которое он должен проехать 10 км.) и прогнозируемое время бега (время, за которое он пробежит 3 км.). Ваш друг хочет спланировать расписание триатлона: порядок, в котором участники будут выходить на дистанцию. Завершающим временем расписания будет называться самое меньшее время, за которое все участники пройдут все три стадии триатлона, при условии, что они точно выдержат спрогнозированное время во всех трех стадиях (еще раз: участники могут ехать на велосипеде и бежать одновременно, но в бассейне в любой момент времени может находиться только один человек).

Как выглядит оптимальный порядок выхода участников, чтобы соревнования закончились как можно раньше? А точнее, предложите эффективный алгоритм, который строит расписание с минимальным завершающим временем.

Упражнение 21

Решите, истинно или ложно следующее утверждение. Если оно истинно, приведите краткое объяснение, а если ложно — контрпример.

- Пусть G — произвольный связный ненаправленный граф с различающимися стоимостями c(e) всех ребер e. Предположим, e* — ребро с минимальной стоимостью в G, то есть c(e*) < c(e) для каждого ребра e ≠ e*. Тогда существует минимальное остовное дерево T графа G, содержащее ребро e*.

Упражнение 22

Решите, истинно или ложно каждое из следующих двух утверждений. Если утверждение истинно, приведите краткое объяснение, а если ложно — контрпример.

(a) Предположим, имеется экземпляр задачи нахождения минимального остовного дерева для графа G, все стоимости ребер которого положительны и различны. Пусть T — минимальное остовное дерево для этого экземпляра.

Стоимость каждого ребра c(e) заменяется его квадратом $c^2(e)$; так образуется новый экземпляр задачи с тем же графом, но другими стоимостями.

Истинно или ложно? Дерево T должно остаться минимальным остовным деревом для нового экземпляра задачи.

(b) Предположим, имеется экземпляр задачи нахождения кратчайшего пути s–t в направленном графе G. Все стоимости ребер положительны и различны. Пусть P — путь s–t для этого экземпляра с минимальной стоимостью.

Стоимость каждого ребра c(e) заменяется его квадратом $c^2(e)$; так образуется новый экземпляр задачи с тем же графом, но другими стоимостями.

Истинно или ложно? Путь P должен остаться путем s–t с минимальной стоимостью для нового экземпляра задачи.

Упражнение 23

Одна из практических целей для задачи нахождения минимального остовного дерева — построение остовной сети для множества узлов с минимальной общей стоимостью. Однако сейчас нас интересует другой тип целей: проектирование остовной сети, в которой максимальная стоимость ребра настолько низка, на сколько это возможно.

Говоря конкретнее, пусть $G = (V, E)$ — связный граф с n вершинами, m ребрами и положительными стоимостями ребер (предполагается, что все стоимости различны). Пусть $T = (V, E')$ — остовное дерево G; *критическим ребром* дерева T будет называться ребро из T с наибольшей стоимостью.

Остовное дерево T графа G называется *минимально-критическим остовным деревом* (minimum bottleneck spanning tree), если не существует остовного дерева T' графа G с меньшей стоимостью критичного ребра.

(a) Является ли каждое минимально-критичное дерево графа G минимальным остовным деревом G? Докажите или приведите контрпример.

(b) Является ли каждое минимальное остовное дерево графа G минимально-критичным деревом графа G? Докажите или приведите контрпример.

Упражнение 24

Пусть $G = (V, E)$ — (ненаправленный) граф со стоимостями $c_e \geq 0$ ребер $e \in E$.

Допустим, вам известно минимальное остовное дерево T для G. Предположим, что в G добавляется новое ребро, соединяющее два узла $v, w \in V$ со стоимостью c.

(a) Предложите эффективный алгоритм проверки того, остается ли T остовным деревом минимальной стоимости при добавлении нового ребра в G (но не в дерево T). Алгоритм должен выполняться за время $O(|E|)$. Удастся ли вам заставить его выполняться за время $O(|V|)$? Отметьте все предположения, которые вам пришлось сделать относительно структуры данных, используемой для представления дерева T и графа G.

(b) Допустим, T уже не является минимальным остовным деревом. Предложите алгоритм с линейным временем ($O(|E|)$) для замены T новым минимальным остовным деревом.

Упражнение 25

Предположим, имеется связный граф $G = (V, E)$, ребро e которого имело стоимость c_e. При различных стоимостях ребер G содержит уникальное минимальное остовное дерево. Но если в графе имеются ребра с одинаковой стоимостью, G может содержать несколько минимальных остовных деревьев.

Вопрос: можно ли заставить алгоритм Крускала найти все минимальные остовные деревья графа G?

Вспомните, что алгоритм Крускала сортирует ребра в порядке увеличения стоимости, а затем жадно обрабатывает ребра одно за другим, добавляя ребро e при условии, что оно не приводит к образованию цикла. Когда некоторые ребра имеют одинаковую стоимость, выражение «в порядке увеличения стоимости» необходимо задать более тщательно: упорядочение ребер называется *действительным*, если соответствующая последовательность стоимостей ребер не уменьшается. *Действительным выполнением* алгоритма Крускала будет называться выполнение, начинающее с действительного упорядочения ребер G.

Для любого графа G и любого минимального остовного дерева T графа G существует ли действительное выполнение алгоритма Крускала для G, которое выдает T как результат? Приведите доказательство или контрпример.

Упражнение 26

Вы работаете главным логистом в транспортной компании «Быстрый Верблюд». В вашем регионе действует сеть торговых путей, связывающая n оазисов.

Каждая дорога находится под охраной одного из двух влиятельных бедуинских племён: «Аназа» или «Бани Яс». Для обеспечения безопасности грузов и оптимизации расходов на охрану, компания хочет выбрать набор дорог, образующий связную систему маршрутов без замкнутых петель, проходящую через все n оазисов.

Директор компании согласовал с племенами квоту: k дорог в выбранной сети должны контролироваться племенем «Аназа», а остальные $n-k-1$ дорог — племенем «Бани Яс».

Руководство столкнулось с неопределённостью: непонятно, существует ли вообще конфигурация дорог, удовлетворяющая этому требованию, и как её найти, если она существует.

Требуется разработать алгоритм с полиномиальным временем работы, который по заданной карте дорог и распределению охраны либо

1) возвращает список дорог, образующих требуемую сеть ровно с k дорогами под охраной племени «Аназа», либо

2) обоснованно сообщает, что выполнить условие невозможно.

Упражнение 27

Имеется массив A из n элементов, в которых хранятся несовпадающие числа. Известно, что последовательность значений A[1], A[2], ..., A[n] унимодальна: для некоторого индекса p в диапазоне от 1 до n значения в элементах массива возрастают до позиции P, а затем убывают до позиции n. (Если нарисовать график, на котором ось x представляет позицию элемента в массиве j, а ось y — значение элемента A[j], график будет возрастать до точки p на оси x, где достигается максимум, а затем будет непрерывно убывать.)

Требуется найти «пиковый элемент» p без чтения всего массива, более того, нужно ограничиться чтением минимально возможного количества элементов A. Покажите, как найти p чтением не более O(log n) элементов A.

Упражнение 28

Вы работаете на небольшую инвестиционную компанию, проводящую интенсивную обработку данных; в вашей работе приходится снова и снова решать задачу определенного типа.

Типичная ситуация выглядит так. Проводится моделирование, при котором рассматриваются курс акций за n последовательных дней где-то в прошлом. Пронумеруем дни $i = 1, 2, ..., n$; для каждого дня i известна биржевая котировка этих акций $p(i)$ в этот день. (Для простоты будем считать, что в течение дня котировка не изменяется.) Предположим, где-то в этот период времени нужно купить 1000 акций в один день и продать все эти акции в другой (более поздний) день. Вопрос: когда покупать (и когда продавать) акции для получения максимальной прибыли? (Если за эти n дней перепродажа с прибылью невозможна, алгоритм должен сообщить об этом.)

Предположим, $n = 3$, $p(1) = 9$, $p(2) = 1$, $p(3) = 5$. Алгоритм должен вернуть «Покупать в день 2, продавать в день 3» (в этом случае прибыль составит $4 на акцию — максимально возможная прибыль за этот период).

Очевидно, существует простой алгоритм с временем $O(n^2)$: опробовать все возможные комбинации дней покупки/продажи и посмотреть, какой из них обеспечивает максимальную прибыль. Ваши работодатели надеются иметь более эффективное решение.

Упражнение 29

Вы управляете строительством рекламных щитов на дороге с интенсивным движением, идущей с запада на восток на протяжении M км. Возможные точки для установки рекламных щитов определяются числами $x_1, x_2, \ldots, x_n$, каждое из которых принадлежит интервалу $[0, M]$ (эти числа определяют позицию в км. от западного конца дороги). Разместив рекламный щит в точке x_i, вы получите прибыль $r_i > 0$. По правилам дорожной службы округа два рекламных щита не могут находиться на расстоянии 5 км. и менее. Требуется разместить рекламные щиты на подмножестве точек так, чтобы обеспечить максимальную прибыль с учетом ограничения.

Пример. Предположим, $M = 20$, $n = 4$,

$(x_1, x_2, x_3, x_4) = (6, 7, 12, 14)$, и $(r_1, r_2, r_3, r_4) = (5, 6, 5, 1)$.

В оптимальном решении рекламные щиты размещаются в точках x_1 и x_3, с суммарной прибылью 10.

Предложите алгоритм, который получает экземпляр этой задачи и возвращает максимальную суммарную прибыль, которая может быть получена для любого действительного подмножества точек. Время выполнения алгоритма должно быть полиномиальным по n.

Упражнение 30

Имеется ненаправленный граф с n узлами $G = (V, E)$. Напомним, что подмножество узлов называется *независимым множеством*, если никакие два узла не соединяются ребром. В общем виде задача нахождения большого независимого множества сложна, но при достаточно «простом» графе она имеет эффективное решение.

Назовем граф $G = (V, E)$ *путевым графом*, если его узлы могут быть записаны в виде $v_1, v_2, ..., v_n$, причем v_i и v_j соединяются ребром в том и только в том случае, если i и j различаются ровно на 1. С каждым узлом vi связывается положительный целый *вес* w_i.

Целью этого упражнения является решение следующей задачи: Найдите в путевом графе G независимое множество с максимальным общим весом.

Упражнение 30(а)

(а) Приведите пример, показывающий, что следующий алгоритм *не всегда* находит независимое множество с максимальным общим весом.

Жадный алгоритм «приоритетного выбора максимального веса»

Начать с S, равного пустому множеству

Пока в G остаются узлы

    Выбрать узел v_i с максимальным весом

    Добавить v_i в S

    Удалить v_i и его соседей из G

Конец Пока

Вернуть S

Упражнение 30(b, c)

(b) Приведите пример, показывающий что следующий алгоритм тоже *не всегда* находит независимое множество с максимальным общим весом.

- Присвоить S_1 множество всех vi, для которых i нечетно
- Присвоить S_2 множество всех v_i, для которых i четно (обратите внимание: S_1 и S_2 являются независимыми множествами)
- Определить, какое из множеств S_1 или S_2 имеет больший общий вес, и вернуть его.

(c) Предложите алгоритм, который получает путевой граф G из n узлов с весами и возвращает независимое множество с максимальным общим весом. Время выполнения должно быть полиномиальным по n независимо от значений весов.