owatun015
2 июн 2024 в 01:36

Механика и стратегия игры «5букв»

Простой 7 мин 8K

Алгоритмы*Логические игры

Введение

Игры, в которых нужно отгадывать слова достаточно популярны. Игра *5букв* одна из таких игр. Особенность данной игры заключается в том, что нужно угадать русское существительное из 5 букв. На это дается 6 попыток. В качестве попытки может быть использовано любое русское существительное из 5 букв. Несуществующие слова использовать нельзя.

После каждой попытки буквы окрашиваются в следующие цвета.

- Серый – означает, что буквы нет в слове.
- Белый – означает, что буква есть в слове, но её нет в данной позиции.
- Жёлтый – означает, что буква есть в данной позиции слова.

Очевидно, что если слово угадано, то все буквы окрашиваются в жёлтый цвет.

Буква *Ё* в игре не используется, вместо неё используется буква *Е*.

Далее ситуации соответствующие цветам будем просто обозначать этими цветами.

Подготовка и постановка цели и задач

В первую очередь поищем словарь русских существительных, оставим в нём только пятибуквенные слова, заменим букву *Ё* на букву *Е*, и удалим дубликаты (они могут появится как минимум в результате замены *Ё* на *Е*). Слова в словаре отсортируем.

Информация о внутренностях игры у нас отсутствует. Взаимодействовать с ней можно только через пользовательский интерфейс. Поэтому для полноты картины сделаны следующие предположения.

- Игра использует слова из найденного словаря.
- Игра не сопоставляет слову какой-либо *вес* (например: популярность слова). Выбор каждого слова из словаря равновероятен.
- Слова между играми могут повторяться.

После обработки вышеуказанного файла получилось 2068 слов.

Примечание: по каким-то причинам исходный словарь не содержал следующие слова *кошка*, *скула*, *хмель*. Эти слова добавлены вручную.

Цель и задачи:

- Придумать стратегию игры, позволяющую выигрывать с вероятностью 99,9% в рамках полученного словаря.
- Интерактивное консольное приложение для взаимодействия с игрой. Игра предлагает слово для ввода, а пользователь вводит ответ игры.

Идея решения

В качестве хода в игре используется слово. Чтобы понять какое слово использовать, необходимо ввести его числовую метрику.

Пусть словарь w_k содержит $n, n > 0$, которые уже удовлетворяют требованиям предыдущих $k - 1$ попыток. Пусть некоторое слово с индексом $j, j = \overline{0; n - 1}$ выбрано игрой (далее: слово j). Пусть игрок делает предположение некоторого слова с индексом $i, i = \overline{0; n - 1}$ (далее: слово i). Сколько слов останется, если выбрано слово i, а игра загадала слово j? Моделирование процесса позволит отфильтровать слова из словаря w_k для любой пары (i, j).

В рамках одного слова i, но всех возможных слов j можно найти количество оставшихся слов для каждой пары (i, j) и, получить их среднее арифметическое. Это и будет нашей метрикой, которую будем называть *метрикой совпадений*. Чем ближе метрика к 0 – тем лучше.

Вычисляя метрику совпадений для всех слов i, находим такое, у которого эта метрика минимальна. Таких слов может быть несколько, выбрать можно любое.

Реализация

Поиск лучшего кандидата

Сердце реализации – это функция получения лучших предположений. В ней действительно будет 3 вложенных цикла. Для всего словаря необходимо $2068^3 = 8844058432$ операций по выяснению удовлетворения словом ограничений. Возможно цифра покажется большой, но

(забегая вперед) даже программирование “в лоб” позволяет найти лучшего кандидата примерно за 2 минуты.

Тем не менее попробуем ещё ускорить. Заметим, что в рамках одной попытки фигурирует один и тот же набор слов – словарь w_k . При поиске минимума используется сравнение среднего арифметического, при вычислении которого последняя операция это всегда деление на одно и тоже число – размер словаря, т. е. $\left|w_k\right|$. Поэтому среднее арифметическое можно заменить простой суммой (как бы умножив его снова на размер словаря). Что это даст? Во-первых: никаких чисел с плавающей точкой. Во-вторых, теперь мы можем поймать момент когда оценка для текущего предположения стала больше, чем хранимый текущий минимум. Это означает, что предположение точно хуже и поэтому отбросить его можно “досрочно”. Чем меньше становится минимум – тем быстрее поиск.

Каждое слово может быть загадано равновероятно и никакой другой информации до первого предположения у нас нет. Таким образом, начальное состояние игры с точки зрения игрока всегда одинаково. Поэтому и первое слово, которые вводится в игру должно быть всегда одно и то же. Будет разумным вычислить его заранее.

Структуры данных для ограничений

Для хранения ограничений будем использовать два массива.

- Массив состояний каждой буквы алфавита. Каждый элемент может принимать одно из следующих значений: *неизвестно*, *обязательно присутствует в слове*, *обязательно отсутствует в слове*. Размер: 32 элемента. Далее сам массив будем называть *массивом букв*.
- Массив состояний каждой позиции. Каждый элемент может принимать следующие значения: *неизвестно*, *известно*. В состоянии *известно* дополнительно хранится информация о какой букве идет речь и признак присутствия (жёлтая ситуация) или отсутствия (белая ситуация) этой буквы на позиции. Размер: 5 элементов. Далее сам массив будем называть *массивом позиций*. **Массив позиций располагает информацией только о жёлтых и белых ситуациях.**

Совокупность этих двух массивов будем называть *состоянием ограничений* или просто *состоянием*.

В интерактивном режиме эти два массива будут получены из *маски*, которую вводит пользователь.

Маска представляет из себя 5 упорядоченных элементов. Каждый элемент это одно из трех значений: серый, белый, жёлтый.

Для того, чтобы всё заработало необходимо написать следующие вспомогательные функции.

- Получение состояния по предполагаемому и загаданному словам (для имитации игры).
- Получение состояния по слову и маске.
- Проверка удовлетворяет ли слово состоянию.

Получение состояния по двум словам.

1. Объявить массив букв, в котором все элементы в состоянии *неизвестно*. Объявить массив позиций, в котором все элементы в состоянии *неизвестно*.
2. Пометить в массиве букв как отсутствующие все буквы предполагаемого слова.
3. Пометить в массиве букв как присутствующие только те буквы загаданного слова, которые уже помечены как отсутствующие.
4. Организовать цикл прохода по массиву позиций. Каждую позицию обновить по следующим правилам.
    1. (Жёлтая ситуация) Если буквы в текущем позиции загаданного слова и предполагаемого слова совпадают, то позиция переходит в состояние *известно* с соответствующей буквой из предполагаемого слова и признаком присутствия.
    2. (Белая ситуация) В противном случае, позиция переходит в состояние *известно* с соответствующей буквой из предполагаемого слова и признаком отсутствия. Сделать это нужно только в том случае, если в массиве букв буква помечена как присутствующая.

Получение состояния по слову и маске.

1. Объявить массив букв, в котором все элементы в состоянии *неизвестно*. Объявить массив позиций, в котором все элементы в состоянии *неизвестно*.
2. Организовать цикла прохода по буквам слова и соответствующим элементам маски параллельно.
    1. Элемент маски серый. Пометить букву в массиве букв как отсутствующую.
    2. Элемент маски белый. Пометить букву в массиве букв как присутствующую. Записать в соответствующий элемент массива позиций *известно* вместе с соответствующей буквой слова и признаком её отсутствия на позиции.
    3. Элемент маски жёлтый. Пометить букву в массиве букв как присутствующую. Записать в соответствующий элемент массива позиций *известно* вместе с соответствующей буквой слова и признаком её присутствия на позиции.

Проверка подходит ли слово под ограничения.

1. Если хотя бы одна из букв слова помечена как отсутствующая в массиве букв – слово не подходит. Конец.
2. Если хотя бы одна из букв не подходит под ограничение соответствующего элемента в массиве позиций – слово не подходит. Конец.
3. Если количество уникальных букв слова помеченных как присутствующие в массиве букв меньше, чем общее количество букв помеченных как присутствующие в массиве букв – слово не подходит. Конец.
4. Достижение этого пункта сигнализирует о том, что слово подходит.

Примечание: в процедуре слияния состояний между попытками нет никакой необходимости, вместо этого используется фильтрация слов.

Режимы работы приложения

Итак, исходя из вышесказанного интересны следующие режимы работы приложения.

- Вычисление первого предположения.
- Интерактивный режим работы с пользователем.
- Сбор полной статистики по всем словам.

Интерактивный режим подразумевает работу следующего сценария.

1. Приложение предлагает слово для ввода в игру.
2. Пользователь вводит это слово в игру. Игра окрашивает буквы в цвета.
3. Пользователь вводит маску в приложение. Маска состоит из 5 символов, где каждый символ это: *y* – цвет буквы жёлтый, *w* – цвет буквы белый, *g* – цвет буквы серый.
4. Программа предлагает следующее слово для ввода в игру и процесс повторяется до тех пор пока пользователь не введёт *yyyyy*.

Сбор полной статистики имитирует раунды игры. Количество раундов соответствует количеству слов в словаре. Каждое слово из словаря загадывается ровно один раз. В процессе определяются следующие величины.

- Максимальное количество попыток.
- Количество проигрышей (и сами загаданные слова).

Результаты

Весь код можно найти тут.

Первое слово, которое нужно ввести в игру: *катер*. Его поиск занимает около минуты времени. Это слово сильно ограничивает количество оставшихся слов, поэтому дальнейший поиск работает практически мгновенно.

Сбор полной статистики тоже занимает около минуты. Два слова, которые не могут быть отгаданы за 6 попыток: *гряда* и *щепка*. Для них требуется 7 попыток. Итого, вероятность выигрыша составляет $2066 \div 2068 = 0,999 = 99,9\%$.

Данная статья не ставит перед собой задачу сделать поиск наиболее оптимальным. Наверняка где-то что-то упирается в локальный максимум и жадность тут описанная, не совсем до конца верное решение. Тем не менее, она работает и неплохо удовлетворяет закону Парето.

Примеры отгаданных слов

Слово						Цвета (реакция игры)				
К	А	Т	Е	Р		Б	Б	С	С	С
С	О	Л	К	А		Ж	С	Б	Б	Ж
С	К	У	Л	А		Ж	Ж	Ж	Ж	Ж

Слово						Цвета (реакция игры)				
К	А	Т	Е	Р		С	С	С	Б	С
С	Е	Л	И	Н		С	Б	Б	С	С
П	Л	Е	М	Я		С	Б	Ж	Б	С
Х	М	Е	Л	Ь		С	Ж	Ж	Ж	Ж
Ш	М	Е	Л	Ь		Ж	Ж	Ж	Ж	Ж

Слово						Цвета (реакция игры)				
К	А	Т	Е	Р		С	Б	С	С	Б
П	У	Р	Г	А		Ж	Ж	Ж	Ж	Ж

Заключение

Статья уже была написана и готова к публикации как вдруг игра загадала слово “канон”. Процесс отгадывания проиллюстрирован на рисунке.

Слово						Цвета (реакция игры)				
К	А	Т	Е	Р		Ж	Ж	С	С	С
К	А	Л	А	Н		Ж	Ж	С	С	Ж
К	А	М	И	Н		Ж	Ж	С	С	Ж
К	А	Н	О	Н		Ж	Ж	Ж	Ж	Ж

Стоит обратить внимание, что на второй попытке одна буква *А* подсвечена желтым, а другая – серым. Правилам игры допускают повторение букв в слове, но как игра такие буквы раскрашивает, они не раскрывают. Всего скорее упрощенный алгоритм раскрытия букв примерно следующий.

1. Раскрыть все совпадающие буквы (жёлтая ситуация) и убрать их из дальнейшего рассмотрения.
2. Для каждой уникальной буквы (из оставшихся) загаданного слова построить ассоциативный массив: *буква* $\rightarrow$ *количество*.
3. Пройтись по всем буквам предполагаемого слова. Для каждой буквы проверить есть ли она в ассоциативном массиве. Если есть, и количество не достигло нуля, то покрасить букву в белый цвет и уменьшить количество на единицу. Во всех остальных случаях покрасить букву в серый цвет.

С точки зрения нашей модели, если хотя бы одна буква окрашена в белый или жёлтый цвета, то и остальные такие же буквы будут окрашены либо в белый, либо в жёлтый цвета.

Очевидно, что игра на самом деле раскрывает еще больше информации о загаданном слове. Больше информации – больше ограничений. Больше ограничений – меньше подходящих слов.

Для того, чтобы модель оставалось простой и всё ещё работающей с игрой введено два простых правила преобразования серых ситуаций при вводе маски пользователем.

- Помечаем букву в массиве букв как отсутствующую только тогда, когда в массиве букв она не помечена присутствующей.
- Обработка серой ситуации дополняется обновление массива позиций аналогично белой ситуации.

Таким образом, ранее рассмотренные серые буквы могут стать белыми и для этого нужно только обновить массив букв. Буква, став однажды белой, остается белой “навсегда”.

Итак, модель допускает немного больше кандидатов, чем игра и поэтому продолжает быть полезной.

P.S

Вот в этом комментарии рассказано как можно докрутить метрику и сделать вероятность разгадывания 100%. Новый словарь содержит 3477 слов. Первое слово: норка.

Теги: 5букв5bukvc#algorithms

Хабы: АлгоритмыЛогические игры