АВТОМАТИЧЕСКАЯ СИСТЕМА АНАЛИЗА СПИСКОВ АБИТУРИЕНТОВ

Щагин Е. А., ученик 11 класса АГ СПбГУ, evgenii@eulerlabs.ru

Аннотация

Данная работа посвящена исследованию процесса поступления на программы бакалавриата и специалитета Санкт-Петербургского государственного университета с целью создания автоматических средств мониторинга и модерации списков абитуриентов.

В ходе работы был создан программный продукт, позволяющий дать более точную оценку шансов поступления абитуриента на конкретную программу.

Введение

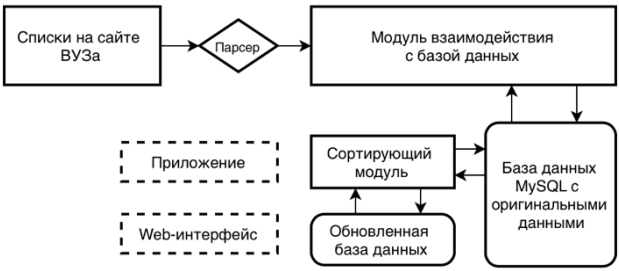
Подача заявлений на программы высшего образования является одним из важнейших шагов на пути абитуриента. Правильно выбранные направления и приоритеты заметно повышают шансы поступающего на зачисление.

Несмотря на наличие самых современных систем учета, на данный момент абитуриенты вынуждены вручную отслеживать все изменения в списках, постоянно происходящие на всем протяжении приема. В зависимости от изменения позиции в рейтинге, поступающий изменяет приоритеты или подает заявления на другие программы, вероятность поступления на которые для него будет выше. В свою очередь, это приводит к изменению рейтингов всех остальных и так далее.

При этом иногда бывает очень затруднительно оценивать свое положение вручную. Это связано с тем, что абитуриенты, подающие заявления на несколько программ внутри одного вуза, занимают несколько мест в разных списках, но в итоге пойдут только на одно направление. В таком случае, освободившиеся места займут другие поступающие, и весь список сдвинется на одну позицию вверх. Это весьма важно для абитуриентов, занимающих позиции, близкие к численной границе приема на конкретное направление.

Описание модели

Система состоит из нескольких самостоятельных модулей, каждый из которых выполняет конкретную функцию. Логику взаимодействия модулей иллюстрирует следующая схема.



Во время проектирования был сделан упор на максимальную надежность системы и был выбран более простой подход к проектированию, но в то же время сохранена возможность масштабирования системы в будущем. Так, например, чтобы подключить к системе новые ВУЗы, достаточно написать парсеры для обработки их данных.

Ввиду широкой распространенности, кроссплатформенности и наличия подробной документации, в качестве инструментов реализации системы были выбраны язык программирования Java [1][2] и СУБД MySQL.

Обзор

Главной функцией системы является модерация списков абитуриентов в соответствии с правилом:

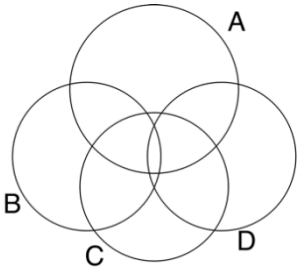
> *Для каждого направления удалять из списка абитуриентов, точно проходящих на другое направление по более высокому приоритету.*

Используя это правило, мы не получим абсолютно точного результата, поскольку существует еще несколько незначительных коллизий, не решаемых данным алгоритмом. Тем не менее, мы сможем получить достаточно точное приближение.

Для того, чтобы продемонстрировать идею алгоритма, можно построить следующую диаграмму. Пусть A – таблица, над которой мы работаем, B, C, D – абитуриенты, проходящие на другие направления по более высокому приоритету. Тогда после модерации в списке должны остаться абитуриенты:

A \ B \ C \ D

Примечательно, что один проход подобным алгоритмом по всем спискам не даст желаемого результата. После обновления, когда все списки сдвинутся вверх, студенты, первоначально не проходившие по первому приоритету, могут оказаться зачисленными. В таком случае освободятся другие места, и операцию придется повторить заново. Более того, для получения наиболее точного результата операцию придется повторять до тех пор, пока после очередной итерации списки не перестанут изменяться.



Была исследована зависимость времени выполнения операции вычитания от количества пересечений.

Количество пересечений	Время, с
2	0,038
16	0,177
24	1,344
26	4,719
28	18,125
29	46,172
30	71,375

Приближение полиномом Лагранжа показало, что данная зависимость имеет асимптотику порядка n^3. Оказалось, что запросы для малого числа пересечений (до двадцати) работают значительно быстрее. Поскольку все программы имеют примерно одинаковое количество поступающих, было решено дробить запросы на несколько подзапросов, по 20 пересечений в каждом, время выполнения которых можно считать константой.

$$\left(\frac{n}{20}\right) * const \ll n^3$$

Заключение

Результатом данной работы является рекомендательная система, модерирующая списки абитуриентов для получения более точной оценки шансов поступления абитуриента на конкретную программу высшего

образования. В то же время, данная система может быть легко адаптирована к любым другим возможным применениям, в которых необходимы мониторинг и модерация списков очередности.

Для увеличения производительности системы возможно изучение алгоритмов внутренней оптимизации MySQL [3] с целью формирования более точных выборок и получения оптимальных оценок. Также возможна многопоточная реализация, позволяющая распараллеливать процесс поиска пересечений.

В качестве возможного продолжения развития системы может быть предложено создание онлайн платформы или мобильного приложения, облегчающих доступ абитуриентов и членов приемных комиссий к отсортированным спискам.

Список литературы

1. Eckel Bruce Thinking in Java [Книга]. - [б.м.]: Prentice Hall PTR, 2002. - 3rd.
2. Reese George Database programming with JDBC and Java [Книга]. - [б.м.]: O'Reilly, 2000. - 2nd.
3. Balling Derek J. High Performance MySQL [Книга]. - [б.м.]: O'Reilly, 2004.