1 Состояние MGRA-сервера до начала стажировки:

1. MGRA-сервер корректно работал на одном из четырех примеров.
2. Данные, загружаемые пользователем, не давали корректного результата.
3. Интерфейс был сыроват и непонятен для работы пользователей.
4. Деревья имели выделенные корни, которые биологического смысла не несли.
5. Содержались ошибки в отображении информации о геноме и трансформации (номера хромосом, спецсимволы).

2 Действия в течение двух месяцев:

1. Были прочитаны статьи и получено представление об области в целом.
   http://genome.cshlp.org/content/19/5/943.full.pdf
   http://genome.cshlp.org/content/13/1/37.short
   http://www.biomedcentral.com/1471-2164/13/129
2. Было добавлено три новых примера с данными для работы с MGRA.
3. Были найдены и исправлены баги, чтобы сервер заработал на всех, теперь уже шести-семи, примерах.
4. Представление деревьев было переделано с таблиц на двоичные деревья.
5. Был убран корневой корень и добавлено преобразование трансформаций.
6. Была добавлена возможность отображения геномов в виде изображений с учетом размера, если блоки введены в формате «infercars».
7. Была добавлена возможность отображения перестановок из трансформаций в виде изображения.
8. Была добавлена возможность отображения реконструированных деревьев на основе данных, которые выдал mgra tool.
9. Была добавлена возможность отображения текущих введенных деревьев.
10. Было добавлено небольшое количество проверок на валидность введенных данных.
11. Была добавлена возможность загрузки полученных текстовых файлов с геномами и трансформациями, выданными mgra tool.
12. Была добавлена возможность загрузки конфигурационного файла и файла с исходными геномами.
13. Была добавлена возможность загрузки полученных изображений с сервера.
14. Была изменена логика работы сервера.
    До этого обработка всех данных происходила один раз, и на выходе генерировался статический html.
    Теперь на первом этапе происходит генерация страницы, отображающей дерево. Если существует информация о геномах и трансформациях, соответствующие узлы и ребра дерева подсвечиваются. При нажатии на них происходит событие, которое приводит к генерации соответствующей информации.
15. Был полностью переписан интерфейс страницы для ввода данных.
16. Было начато написание отчета для Максима Алексеева.

3 Результат:

MGRA server представляет интерфейс для работы с mgra tool.

3.1 Interface

MGRA server, так же как и mgra tool, работает с двумя форматами представления геномов: GRIMM и INFERCARS. Пользователь может выбрать, в каком формате он будет вводить информацию и воспользоваться соответствующим пунктом меню. В зависимости от формата, ему будет представлена своя форма для введения информации(см рис). Кнопкой "add genome"можно увеличивать количество геномов. Если же пользователь загружает файл, то сервер сам определит формат, останется ввести только уникальные однобуквенные имена и другие названия геномов.

По кнопке "show tree"пользователь может увидеть в новом окне филогенетические деревья, отрисованные с помощью HTML5 CANVAS. Также можно реконструировать деревья (об этой функции написано ниже). В поле "target"и "completion"он может указать вспомогательную информацию, необходимую для работы mgra tool.

Кроме того, на сайте представлен раздел examples, который призван разрешить возможные вопросы пользователей касаемо ввода информации, работы сервера, а также выходных данных. У каждого примера есть краткое пояснение о входных данных и ссылки на соответсвующие статьи, если пользователь захочет узнать о данном исследовании подробно.

3.2 Web-server

Ядро сервера написанно на java с использованием библиотек Jetty для обслуживания запросов пользователя. При нажатии пользователем кнопки run mgra посылается запрос на генерацию данных и отображения их ввиде html. Данные для каждого запроса хранятся в папке, которая соответствует дате и id запроса, например 2011/10/25/request7. В процессе обработки запроса генерируются входные файлы (.cfg и genome.txt), mgra tool запускается отдельным процессом и ожидается его завершение. После этого создается основная страница. На динамическую html-страницу в процессе обработки запроса записывается информация о текущем шаге и возникающих ошибках, а так же всё, что вывел на консоль mgra. Далее происходит десятисекундная задержка, чтобы пользователь успел дочитать инфомацию на странице, и происходит переход к странице с данными.

3.3 Visualizer

MGRA выводит результат в текстовые файлы, а нам нужно представить их в html-формате. Генерация HTML происходит в два этапа:

1. Генерируем XML-документ, используя JDOM
2. Выполняем XSLT-преобразование, используя Saxon

3.4 Tree representation

Ядром всей визуализации является страница, содержащая поддеревья. В первую очередь, здесь пользователь может загрузить входной конфигурационный файл и файл, содержащий исходные геномы, для того чтобы в дальнейшем снова не вводить эти же данные вручную. Далее, в секции "Input subtree(s)"представлены (под)деревья, которые ввел пользователь. Секция "Reconstructed subtree(s)"доступна, если пользователь поставил галочку напротив поля "reconstructed tree". В ней представлены (под)деревья, реконструированные из ветвей, которые находились в файле stats.txt.

Каждое дерево представляется с помощью HTML5 CANVAS с использованием библиотеки KineticJs.

Узлы дерева можно перемещать на любое место в области, где отрисовано дерево. Для этого нужно кликнуть интересующий узел и перетащить его в нужное место.

Помимо этого, пользователь может сохранить дерево в формате .png на своем компьютере. Это можно сделать, нажав кнопку "Save as image".

Если узел дерева окрашен в темно-голубой цвет, то по двойному клику на него пользователь вызовет ajax-запрос серверу для генерации информации о геноме этой вершины. При этом будет выведено сообщение о том, что запрос обрабатывается. В случае успеха, пользователю будет выдана информация о геноме, в случае неудачи - соответствующее сообщение.

Если же ветвь дерева окрашена в черный цвет, то по двойному клику на нее пользователь вызовет ajax-запрос серверу для генерации трансформации между вершинами. Сообщения в этом случае будут аналогичными.

Для удобства работы с ajax-запросами используется библиотека JQuery.

3.5 Genome representation

После того как пользователь вызвал соответствующий запрос на генерацию представления генома, MGRA server пытается сгенерировать изображение, но если возникает какая-либо проблема (картинка большая, занимает много памяти и пр.), то сервер пытается сгененировать xml-документ, а потом с помощью XSLT-преобразования создать html-представление. О представлениях геномов в каждой из сиутаций написано ниже.

1. **Представление в виде изображения**

   (a) infercars
   Плюс этого формата, то что мы знаем длину блоков. Поэтому при отображении в виде изображения мы учитываем разную длину блоков. Но, так как длина такого изображения может быть колоссальна, мы используем следующие трюки:

   i. Находим длину каждого блока в процентах. Это отношение длины блока к длине самой длинной хромосомы.
   ii. Полагаем, что p = 0.85% - это некая константа c, задающая минимальный размер блока. При данном проценте большинство изображений еще влезает в допустимые размеры
   iii. Все блоки, длина которых меньше значения p, отрисовываем с длиной c.
   iv. Все блоки, которые больше, маштабируем, в зависимости от длины, в процентах.
   v. В каждый блок записываем его длину в мега- и кило- величинах.

   пример приведен на рисунке 1

   (b) grimm
   Геном представлен в виде последовательности synteny-блоков. Каждый блок имеет свой номер и ориентацию. Чтобы отображать ориентацию, мы используем пятиугольники, внутри которых записан номер текущего блока. пример приведен на рисунке 2.

2. **Представление в виде html-текста**
   Геном представлен в виде последовательности synteny-блоков. Каждый блок имеет свой номер, кроме того, они ориентированы. Если использовать числа для их описания, будет довольно сложно выяснить длину генома или его частей. Поэтому мы используем знаки $>$ и $<$, в зависимости от их ориентации, для идентифицирования synteny-блоков. Номер блока является заголовком к ссылке на знаки $>$ или $<$.

3.6 Genome transformation representation

После того как пользователь вызвал соответствующий запрос на генерацию трансформации генома, MGRA server старается сгенерировать соответствующее представление трансформации. Причем сервер знает только информацию о концах разрывов трансформации, поэтому возникает проблема ее восстановления. Решаем следующим образом:

1. Разрезаем хромосому на каждом конце трансформации.
2. Объединяем части хромосом в местах, где у них совпадают концы.

После восстановления трансформации происходит генерация информации по ней в html. Причем напротив каждой перестановки имеется кнопка "Create image; после нажатия на которую серверу пошлется запрос на замену html-текста на соответвующую картинку. О представлении трансформации в зависимости от формата изложенно ниже:

1. **Представление в виде html-текста**

   Пример представления этой трансформации изображен на рисунке 3. Есть две секции, "Before"и "After отвечающие за хромосомы до и после трансформации. Хромосомы имеют такой же вид, как и те, что представлены в виде html при отображении генома.

   Для обозначения концов хромосом, которые участвуют в перестановке, мы используем буквы h и t. Каждая часть имеет свой цвет, что позволяет нам показать, где она находится после перестановки. Мы помечаем тем же цветом соответствующие концы после трансформаций.

2. **Представление в виде изображения**

   Пример представления этой трансформации изображен на рисунке 4. Здесь также есть две секции, "Before"и "After где находятся хромосомы до и после трансформации. Хромосомы изображаются так же, как если бы создавалась информация о геноме в виде изображения.

   Куски хромосом, участвующие в перестановке, обводятся прямоугольником определенного цвета, и это позволяет нам показать, где текущие куски находятся после трансформации.

3.7 Algorithm reconstraction trees

Так как mgra выдает только ветви в файл stats.txt, нам необходимо найти из них те, которые не противоречат входным поддеревьям, а также построить по ним дерево. Мы используем следующий алгоритм:

1. Генерируются ветви из входных поддеревьев.
2. Считываются возможные ветви из файла stats.txt без учета ветвей, помеченных тегом bf.
3. Происходит отбор из возможных совместных и наиболее подходящих ветвей. (См. далее)
4. Если получившиеся множество не пусто, то происходит создание новых поддеревьев из этих и сгенерированных из входных поддеревьев ветвей.
5. Если получившиеся множество пусто, то
6. Удаляются ветви, сгенерированные из входных поддеревьев.
7. Запускается mgra tool без данных о входных поддеревьях.
8. Происходит считывание возможных ветвей из файла stats.txt с учетом ветвей, помеченных тегом bf
9. Происходит отбор из возможных совместных и наиболее подхидящих ветвей.
10. На основе множества происходит создание новых поддеревьев.

Прежде чем говорить о том, как происходит отбор, дадим понятие совместных веток:
две ветки, X+Y и Z+T, совместимы тогда и только тогда, когда одна из частей одной ветки (как множество) является подмножеством или надмножеством одной из частей другой ветки. Например, если X - подмножество или надмножество либо Z, либо T, то ветки совместимы.

Отбор возможных ветвей происходит следующим образом:
идет полный перебор всех совместимых веток из таблицы и текущего отобранного множества. Качество результирующего множества мы оцениваем из суммарного качества выбранных веток (параметр, по которому отсортирована таблица в stats.txt). Так мы находим все подходящие поддеревья, совместимые с данными, и выбираем три наилучших.
Сложность этого алгоритма - 2 в степени количества совместимых веток.

4 Будущее:

1. Избавиться от ввода уникальных имен геномов и начать их генерировать самому.
   Проблема в том, что не понятно, как тогда вводить вершины, которые являются предками двух или более особей, например:
   $$((H, R), DOE)$$
   Идея решения: отойти от Newick-формата и вершины DOE, представлять в виде:
   $$((human, rat), \{dog, opposum, elefant\})$$
2. Идея/пожелание, предложенное биологом Ольгой Бочкаревой.
   Так как бывает, что вводные данные нужно часто менять, хотелось бы иметь личный кабинет для пользователей, чтобы можно было работать со своими загруженными данными несколько раз.
   Проблема в том, что, я не знаю, как вообще это можно разумно реализовать в виде интерфейса.
   Максом Алексеевым была предложена идея загружать cfg-файл, но это спасает нас частично и только на первое время. Так как в результате весь интерфейс оказывается не нужен, и пользователь начинает работать только с текстовым файлом, периодически загружая его на сервер. Однако,возможность загрузки cfg-файла так же необходимо реализовать.
3. Тестирование сервера. Ольга Бочкарова обещала предоставить 4(+1 под вопросом) набора входных данных для сервера и алгоритма.
4. Какие-то другие изменения о которых я не знаю.