

A Browser for Pig Genome Data

Thomas Mailund

January 2, 2004

This report briefly describe the blast and alignment data available at <http://www.daimi.au.dk/~mailund/pig-genome/hits.html>. The report describes how the data was generated, how it is stored in the database, and how it is visualised on the web-pages.

1 Introduction

In the following we describe a pilot project for exploring the evolutionary relationship between man *Homo sapiens*, mouse *Mus musculus*, and pig *Sus scrofa*. By aligning pig-genome contigs on the human and mouse genome, and construct triple-alignments between the three species, our ultimate goal is to gain insight into the evolutionary history of the three species. Among other things, we wish to examine the mutation rate and substitution patterns on the three branches of the tree in Fig. 1.

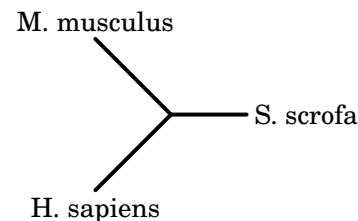


Figure 1: Evolutionary tree for man, mouse, and pig.

The intend of the pilot project was to explore how to place the pig contigs on the two genomes, how to represent the data in a format that enables subsequent analysis, and, to some extend, how to visualise the resulting data in a way that encourage exploratory browsing in the data. In the pilot project we have examined only 70,000 pig sequences; the full project will include around 5 million sequences.

For the human genome, we used assembly hg16 and for the mouse genome we used assembly mm3, both downloaded from UCSC <http://genome.ucsc.edu/downloads.html>. With the rat and chimpanzee genome now also available, these could also be included in the full project.

The report is structured as follows: Section 2 describes how the pig sequences were placed on the human and the mouse genome using the tool megablast. Section 3 describes how the megablast hits together with the human/mouse genome map was used to construct triple-alignments for the three species. Section 4 describes how annotations of the human and the mouse genome was used to annotate the pig sequence hits, Sect. 5 describes how the data is visualised on the web, and finally Sect. 6 draws the conclusions.

2 Blast-hits on *Homo sapiens* and *Mus musculus*

With just the raw pig DNA data, the only way of aligning pig sequences with homologous sequences in the human and mouse genomes is through similarity search. For this, we experimented with three tools: blastz, blastn, and megablast.

Our first choice was the blastz tool, because of its success with aligning the human and the mouse genomes. However, the version of the tool that we used for our experiments was not able to handle databases of sizes of whole genomes, or even whole chromosomes—not on the platform where we conducted the experiments at least. On such large sequences, blastz would just crash.

One way around this problem is to split the genome sequences into fragments, as shown in Fig. 2, and use blastz to search the in the fragments. With a fragment size of 10MB, it took around 30 seconds to search a single fragment for matches to given pig sequence. For the human genome, this means 300 fragments, taking 9000 seconds or 150 minutes (six to seven hours) per sequence. With five million sequences, this is clearly not acceptable. With a fragment size of 100MB, each fragment could be searched in around seven minutes, with a total search time for the human genome around three hours. Still not an acceptable solution.

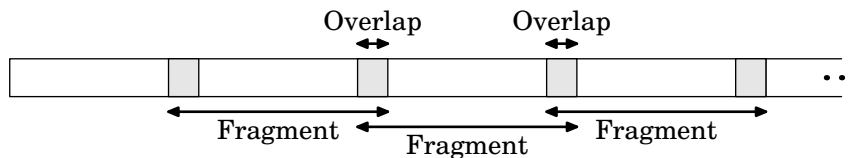


Figure 2: Fragments of a genome for blastz searching. By splitting the genomes into smaller fragments, with large enough overlap to avoid missing hits between the fragments, the genomes could still be searched using blastz.

Since the runtime usage of blastz was unacceptable, we instead chose megablast for positioning the pig sequences on the two other genomes. With megablast it was possible to search an entire genome in one search, and in an acceptable time—one or two minutes for positioning a single pig sequence on the human genome. Most of this time is spend on reading the database into the program, and by batching the searches it is possible to position hundreds of sequences in parallel, with only little increase in runtime.

The algorithm in megablast is a faster search heuristic than that found in blastn, so to make sure that, by choosing the faster tool, we did not miss important hits, we also performed an experiment using blastn. These experiment showed that megablast and blastn found almost the same set of hits (although not exactly the same set, each tool found a few hits not found by the other). The runtime of megablast, however, was superior to that of

blastn. Consequently, megablast was used to place the pig sequences on the human and mouse genomes for the project.

Since we are mainly interested in placing the pig sequences on the two other genomes, we do not use megablast to align the sequences, only to find the positions. We therefore only gather the hit positions and the hit scores—we will align the sequences later in the process, when we have hits on both human and mouse genome, such that we can build a triple alignment between pig, man, and mouse.

We experimented a bit with the options to megablast, to find an acceptable trade-off between runtime and hit-quality. The main parameters to specify are the word-size (`-w`), used to specify the length of an exact match needed to start a local alignment, and the e-value (`-e`), used to filter out low-quality hits. Of these, the word-size option has the main influence on the runtime, whereas the e-value option mainly just lowers the number of hits reported from the tool.

Due to runtime concerns we chose a word-size of 12 for both genomes. The word-size option increases in steps of four, and the next lower word-size, eight, significantly increased the search time. We then wanted to select the e-value option such that as many pig sequences could be placed at a unique location on the two other genomes, while as few pig sequences would have many hits. If a single pig sequence has many hits along the two other genomes, we have no way of determining which is the “right” location, so such sequences would have to be ignored anyway. For the human genome, we chose an e-value of 10^{-30} and for the mouse genome a value of 10^{-15} . The stricter requirement to human hits reflect the closer relationship between the human genome and the pig genome.

The hits found by megablast was stored in a MySQL database, in a table on the form shown below. The table is for the hits on the human genome (`hs`), the table for the mouse genome (`mm`) is similar.

```
CREATE TABLE hs_hits (  
    hit_key          INT UNSIGNED NOT NULL AUTO_INCREMENT,  
  
    ss_key           INT UNSIGNED NOT NULL,  
    ss_start        INT UNSIGNED NOT NULL,  
    ss_end          INT UNSIGNED NOT NULL,  
  
    hs_chromosome   VARCHAR(2) NOT NULL,  
    hs_start        INT UNSIGNED NOT NULL,  
    hs_end          INT UNSIGNED NOT NULL,  
  
    e_value         DOUBLE NOT NULL,  
    bits_score      DOUBLE NOT NULL,  
  
PRIMARY KEY        (hit_key),  
INDEX              ss_key          (ss_key),  
INDEX              hs_chromosome  (hs_chromosome)  
);
```

Even with the filtering on e-values, with the options described above, some pig sequences had more than a few hits along the two genomes. We, arbitrarily, chose a cut-off value of five, and only used pig sequences hitting less than five times on each genome for the further analysis. With this cut-off value we can not uniquely place each pig sequence on the genomes, but restricting the sequences considered to those with only a single hit on each genome would remove most of the hits, and still not guarantee that the pig sequence was placed at the correct location. The cut-off value of five gave us a reasonable number of hits to work with, while still having a limited number of candidate locations for each pig sequence. The filtered sequences were build using the SQL code shown below.

```
CREATE TEMPORARY TABLE tmp_hs_occurrences
    SELECT ss_key,COUNT(*) AS occurrences
    FROM hs_hits GROUP BY ss_key
CREATE TEMPORARY TABLE tmp_mm_occurrences
    SELECT ss_key,COUNT(*) AS occurrences
    FROM mm_hits GROUP BY ss_key
CREATE TEMPORARY TABLE tmp_filtered
    SELECT hs_count.ss_key
    FROM tmp_hs_occurrences as hs_count
    INNER JOIN
        tmp_mm_occurrences as mm_count
    USING (ss_key)
    WHERE hs_count.occurrences <= 5
        AND mm_count.occurrences <= 5
```

3 Triple Alignments

Once the pig sequences are placed on the human and the mouse genomes, the next step is to build triple alignments between the sequences of the three species.

To build the triple alignments, we use the human-mouse map available at UCSC. This map associates fragments of one genome with their homologue fragments on the other genome, as illustrated in Fig. 3. We have stored this map in our database using the table shown below.

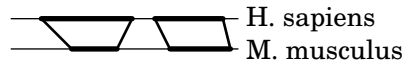


Figure 3: The human-mouse map.

```
CREATE TABLE human_mouse_map (
    map_key          INT UNSIGNED NOT NULL AUTO_INCREMENT,

    hs_chromosome   VARCHAR(2) NOT NULL,
    hs_start        INT UNSIGNED NOT NULL,
    hs_end          INT UNSIGNED NOT NULL,
```

```

mm_chromosome    VARCHAR(2) NOT NULL,
mm_start         INT UNSIGNED NOT NULL,
mm_end          INT UNSIGNED NOT NULL,
PRIMARY KEY (map_key),
INDEX (hs_chromosome,hs_start,hs_end),
INDEX (mm_chromosome,mm_start,mm_end)
);

```

To build a triple alignment, we must have a pig sequence that is located in homologue fragments of the human and the mouse genome. That is, to build a triple alignment we can only use the pig sequences that, by the blast search, have been placed in homologue fragments, as illustrated in Fig. 4.

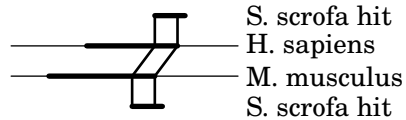


Figure 4: A pig sequence hitting inside a section of the human-mouse map.

To test whether a pig sequence has been located inside a given map-fragment, say the human genome fragment, we test whether the start point of the pig sequence, ss_s , is placed before the end point of the map fragment, hs_e , and the end point of the pig sequence, ss_e , is placed after the start point of the map fragment, hs_s : $ss_s \leq hs_e$ and $hs_s \leq ss_e$, see Fig. 5. This test will place the pig sequence on the fragment if there is the slightest overlap between the two intervals.

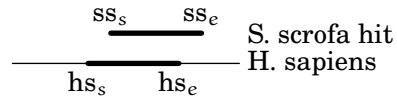


Figure 5: The test for whether a hit is inside a fragment.

In fact, just testing for overlapping intervals is not good enough for finding the candidates for triple alignments; the strands must also be taken into account. There are several cases to consider: the map fragments can be on the same strand or on opposite strands, and the pig sequence can be positioned on either strand of the two other genomes. If the map associates to fragments on the same strand, for the triple alignment to be accepted, the pig sequence must be placed on the same strand of the two genomes; if the map associates fragments on opposite strands, the pig sequence must also be located on opposite strands. The different cases, and the accepted hits, are shown in Fig. 6.

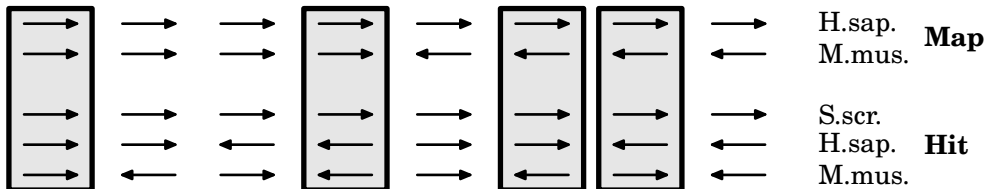


Figure 6: Strand-orientation for the maps and hits. The arrows illustrate the strands—either positive or negative strand. The first four cases correspond to the fragments in the map on the same strand, the last four the cases where the fragments are on the opposite strands. Only the boxed cases, the first, fourth, sixth, and seventh, are accepted for triple alignments.

In our database, we do not store explicitly the triple alignments; instead we store the locations of the sequences to be used in the triple alignments. This lets us experiment with the options for building the alignments from the raw sequences. We store the candidates for triple alignments in a table on the form shown below, that associates a hit on the human genome and a hit on the mouse genome, through the human-mouse map.

```
CREATE TABLE hs_mm_ss_matched_hits (
    hit_key          INT UNSIGNED NOT NULL AUTO_INCREMENT,
    hs_hit           INT UNSIGNED NOT NULL,
    mm_hit           INT UNSIGNED NOT NULL,
    map              INT UNSIGNED NOT NULL,
    PRIMARY KEY (hit_key)
);
```

We build the actual rows by first filtering the hits to eliminate pig sequences with more than five hits on either genome—as described earlier—and then join the hits and map information using the SQL code shown below.

```
INSERT INTO hs_mm_ss_matched_hits (hs_hit,mm_hit,map)
SELECT hs_hit,mm_hit,map_key
FROM tmp_candidates INNER JOIN tmp_hs_candidates AS hs_hits
    ON (hs_hit = hs_hits.hit_key)
    INNER JOIN tmp_mm_candidates AS mm_hits
    ON (mm_hit = mm_hits.hit_key)
    INNER JOIN human_mouse_map AS map
    ON (map.hs_chromosome = hs_hits.hs_chromosome
        AND
        map.mm_chromosome = mm_hits.mm_chromosome)

WHERE
-- hits in the map-intervals
(((hs_hits.hs_start < hs_hits.hs_end)
  AND (hs_hits.hs_start <= map.hs_end)
  AND (map.hs_start <= hs_hits.hs_end))
OR
((hs_hits.hs_start > hs_hits.hs_end)
  AND (hs_hits.hs_end <= map.hs_end)
  AND (map.hs_start <= hs_hits.hs_start))
AND
((map.mm_start < map.mm_end
  AND
  ((mm_hits.mm_start < mm_hits.mm_end)
   AND (mm_hits.mm_start <= map.mm_end)
   AND (map.mm_start <= mm_hits.mm_end))
OR
  ((mm_hits.mm_start < mm_hits.mm_end)
   AND (mm_hits.mm_start <= map.mm_end)
   AND (map.mm_start <= mm_hits.mm_end))))
OR
(map.mm_start > map.mm_end
```

```

AND
((mm_hits.mm_start < mm_hits.mm_end)
 AND (mm_hits.mm_start <= map.mm_start)
 AND (map.mm_end <= mm_hits.mm_end))
OR
((mm_hits.mm_start < mm_hits.mm_end)
 AND (mm_hits.mm_start <= map.mm_start)
 AND (map.mm_end <= mm_hits.mm_end))))))

AND

-- correction directions in map and hits
((map.mm_start < map.mm_end
 AND hs_hits.hs_start < hs_hits.hs_end
 AND mm_hits.mm_start < mm_hits.mm_end)
OR
(map.mm_start < map.mm_end
 AND hs_hits.hs_start > hs_hits.hs_end
 AND mm_hits.mm_start > mm_hits.mm_end)
OR
(map.mm_start > map.mm_end
 AND hs_hits.hs_start < hs_hits.hs_end
 AND mm_hits.mm_start > mm_hits.mm_end)
OR
(map.mm_start > map.mm_end
 AND hs_hits.hs_start > hs_hits.hs_end
 AND mm_hits.mm_start < mm_hits.mm_end))

```

The SQL basically just tests for each of the cases from Fig. 6.

4 Annotations

We have tried annotating the hits, using annotations of the human and the mouse genomes also available from UCSC. For now, we have only included annotations of known genes.

The annotation is stored in a table on the form shown below (for the human genome, a similar table is used for the mouse genome). The annotation contains information about the transcribed parts of genes (tx_start–tx_end), the coding part of genes (cds_start–cds_end), and the exons of the genes (exon_count, exon_starts, and exon_end, stored as a comma separated list of positions).

```

CREATE TABLE hs_known_genes (
  name          VARCHAR(255) NOT NULL,
  hs_chromosome VARCHAR(2)   NOT NULL,
  strand        CHAR(1)     NOT NULL,
  tx_start      INT(10)     UNSIGNED NOT NULL,
  tx_end        INT(10)     UNSIGNED NOT NULL,
  cds_start     INT(10)     UNSIGNED NOT NULL,
  cds_end       INT(10)     UNSIGNED NOT NULL,

```

```

    exon_count          INT(10) UNSIGNED NOT NULL,
    exon_starts        LONGBLOB NOT NULL,
    exon_ends          LONGBLOB NOT NULL,
    protein_id         VARCHAR(40) NOT NULL,
    align_id           VARCHAR(8) NOT NULL,

KEY name (name(10)),
KEY chrom (hs_chromosome,tx_start),
KEY chrom_2 (hs_chromosome,tx_end),
KEY protein (protein_id(12)),
KEY align (align_id),
);

```

We use this information to annotate our hits with information about whether they are in a gene, whether they are in the coding part, and whether they overlap an exon. The hit-annotation is stored in a table on the form shown below.

```

CREATE TABLE hs_gene_annotation (
    hit_key          INT UNSIGNED NOT NULL,
    name            VARCHAR(255) NOT NULL,
    in_coding       INT(1) NOT NULL,
    in_exon        INT(1) NOT NULL,
INDEX(hit_key)
);

```

Testing whether a sequence overlap a gene, the coding region of a gene, or an exon is done similar to the test for whether a sequence overlap a map-fragment.

5 Visualising the Data on the Web

We have built a web-interface to the database, to enable browsing the data. The hope is, that this makes the data more accessible for exploration than raw SQL queries. There are four main views on the data:

- The hit overview page.
- The sequence information pages.
- The alignment information pages.
- The chromosome/annotation pages.

The overview page gives a birds-eye display of the hits along the human and the mouse genomes. For each human and mouse chromosome, it shows the number of hits on the chromosome, the number of triple alignments on the chromosome, and the density of hits and alignments along the chromosome. The page contains links to a detailed description of the hits and annotations along the individual chromosomes, links to pages showing the triple alignments along the individual chromosomes, and links to the UCSC genome browser with a custom track for the pig sequence data. The

links to the UCSC genome browser are placed on the density overviews—where the number of hits and number of alignments are visualised along the chromosomes—and where each segment of the chromosomes contain a link to the genome browser.

The sequence information pages show details for the individual pig sequences. These details include: the actual sequence, the different hits along the human and the mouse genomes, and the triple alignments the sequence is included in.

The alignment information shows either a single alignment or all the alignments along a chromosome. The alignments are built on requests, using the ClustalW multiple alignment tool: the sequences are extracted from a file containing the genomes (for the human and mouse sequences) or the database (for pig sequences), using the indices of the sequences stored in the triple alignment table in the database, and then given to ClustalW. The result is then shown on the web page.

The chromosome/annotation pages shows a list of the hits on a given chromosome (ordered by the positions on the chromosome), together with links to the sequence information, the alignment information (for hits that are part of an alignment), and an annotation of whether the hit is in a gene (and in transcription part, the coding part, or in an exon). For each hit, there is a link to the genome browser, showing the location of the hit.

6 Conclusion

We have conducted a pilot project for the sequenced pig genome data. The project consisted of placing the pig sequences on the human and the mouse genome (using megablast) and building triple alignments between the three species (using the human-mouse map, the megablast hits, and ClustalW), and annotating the hits (using annotations of the human and the mouse genome). We have also built a web interface to the data, making it possible to browse the hits, alignments, and annotation.

Future work now include using the data gathered in the pilot project to determine whether the data is sufficient for the analysis we wish to conduct, whether the browser visualises the right data and in the right format, and so forth.

Once these experiments have been conducted, we will be ready to move to the full project, analysing the full 5 million pig sequences.