UCSC interactive

ucscin.org

rethinking the UI
of genome browsers

Ted Pak
Roth Laboratory
Donnelly Centre, University of Toronto
Samuel Lunenfeld Research Institute,
Mt. Sinai Hospital

motivation

live demo

how it works

motivation

live demo

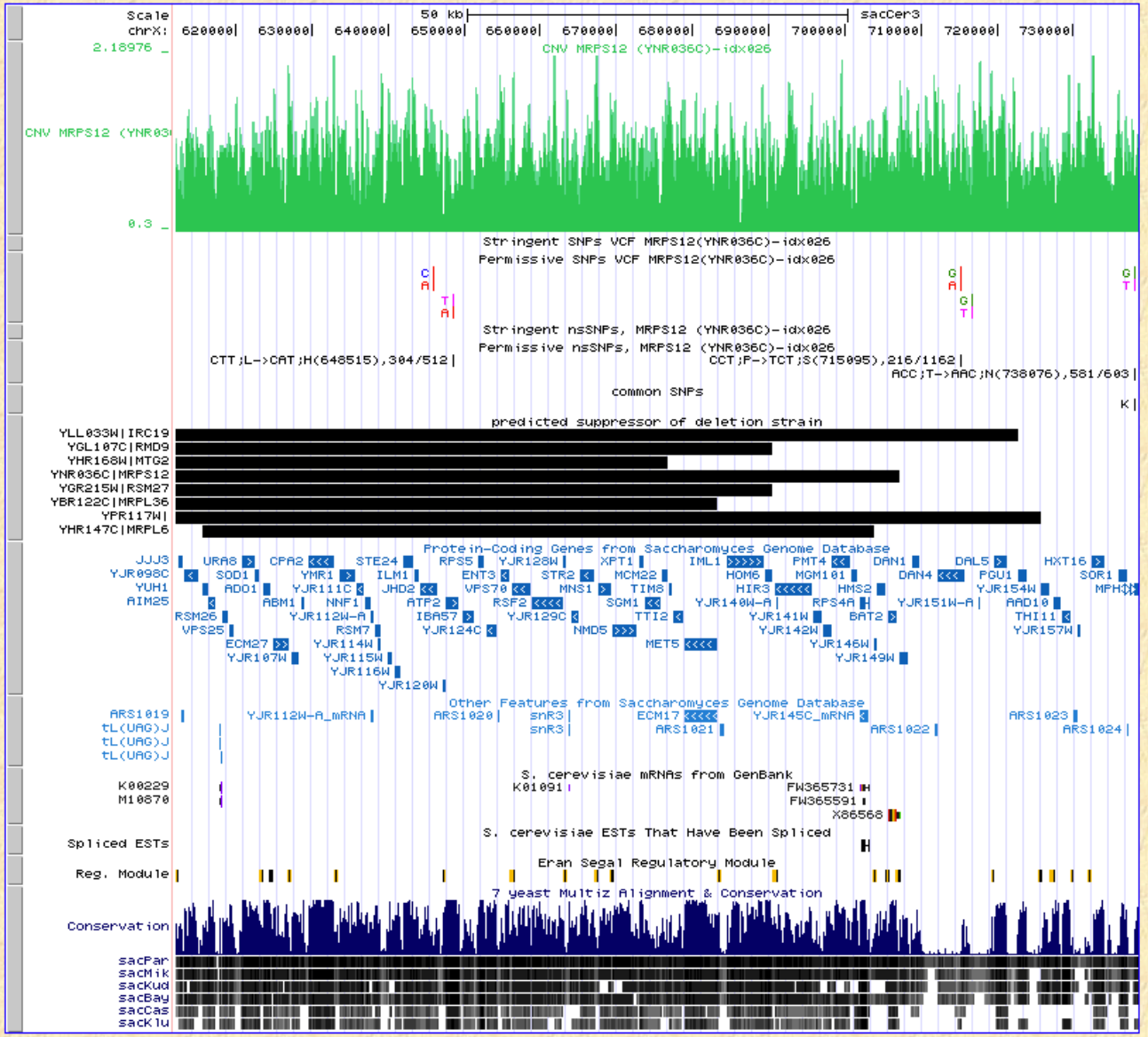
how it works

Home | Genomes | Blat | Tables | Gene Sorter | PCR | DNA | Convert | PDF/PS | Session | Ensembl | Help

UCSC Genome Browser on S. cerevisiae Apr. 2011 (SacCer_Apr2011/sacCer3) Assembly

move <<< << < > >> >>> zoom in 1.5x 3x 10x base zoom out 1.5x 3x 10x

position/search chrX:612,094-738,554 jump clear size 126,461 bp. configure



move start < 2.0 >

Click on a feature for details. Click or drag in the base position track to zoom in. Click side bars for track options. Drag side bars or labels up or down to reorder tracks. Drag tracks left or right to new position.

move end < 2.0 >

Search

Seq Index	Query ORF	Query Gene	Strain ID	Suppressor Locus Link	Mutation in Target Locus?	Notes
idx019	YPL125W	KAP120	Y12760	chrVII:690245-840558	N	
idx020	YFR019W	FAB1	Y12467	chrXIV:734-96080	N	
idx021	YKL139W	CTK1	Y12627	chrXII:951156-1018908	N	
idx022	YGL219C	MDM34	Y12461	chrXII:843-130613	Y	
idx023	YOR330C	MIP1	Y13182	chrII:646-72972	Y	
idx026	YNR036C	MRPS12	Y13177	chrX:581914-708374	N	
idx027	YNL284C	MRPL10	Y13179	chrII:276253-382674	N	
idx028	YNL186W	UBP10	Y13144	chrXI:371829-489655	N	
idx029	YNL177C	MRPL22	Y13161	chrII:646-72972	Y	

Roth lab uses UCSC to

verify hypotheses

inspect specific loci

make figures

but what if I want to

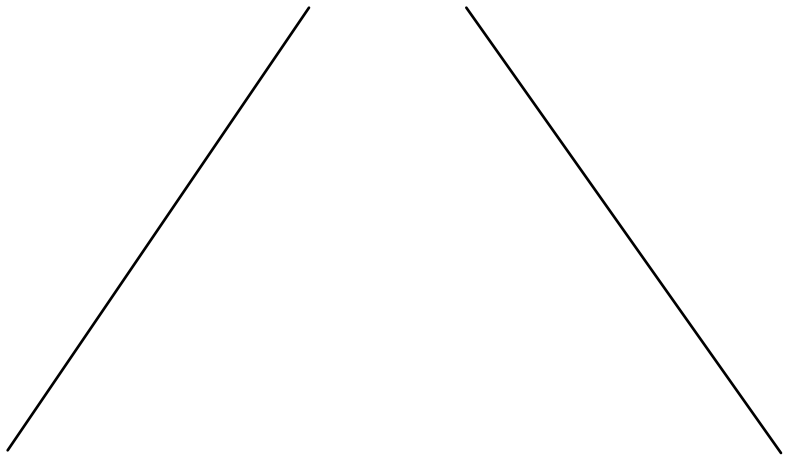
generate hypotheses

explore

discover

the UI problem faced by all genome browsers

lots of data



small viewable area

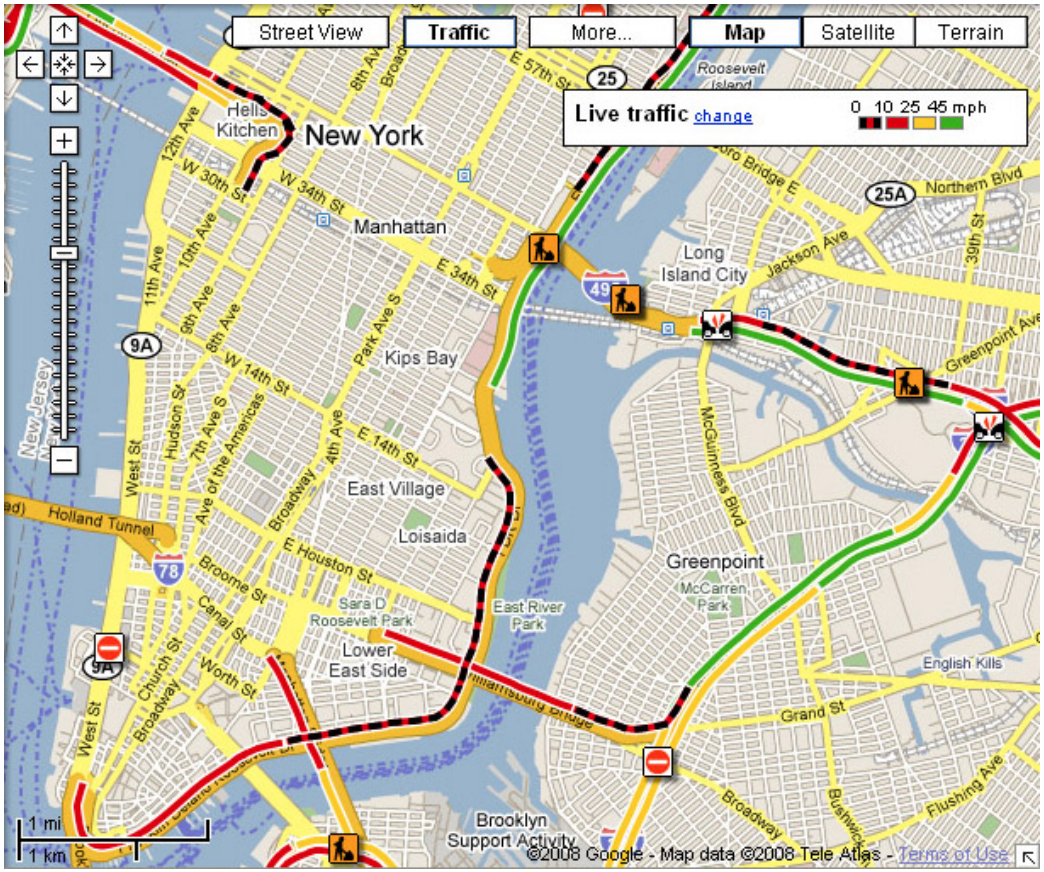
solution 1

reward of solution 1

dangers of solution 1

solution 2


AT&T
7:44 AM
Key
Appointments
Older
Newer
Fri
Dec 18 2009
1:00 PM
Valorie Sherman
Meeting
Mon
Dec 21 2009
09:00 AM
Annie Woo
Interview
12:30 PM
Sherri Knight
Tech Support
Tue
Dec 22 2009
09:30 AM
Annie Woo
Interview
12:30 PM
Jill Wolfe
Meeting
2:00 PM
Valorie Sherman
Meeting
Today
List
Day
Month
Logout

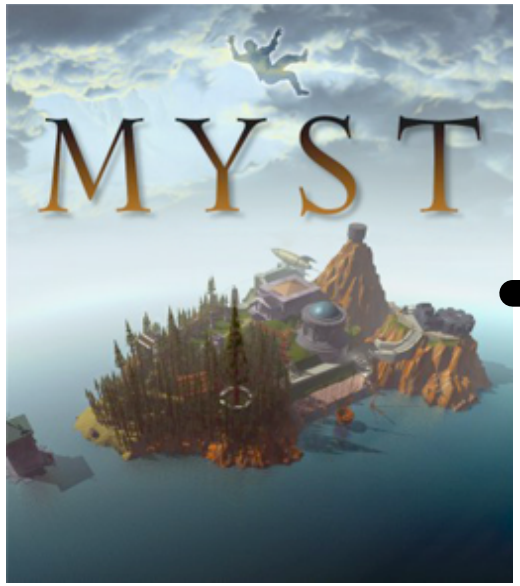

Street View
Traffic
More...
Map
Satellite
Terrain
Live traffic change
0 10 25 45 mph
New York
Manhattan
Long Island City
Kips Bay
East Village
Loisaida
Greenpoint
Lower East Side
Brooklyn
Holland Tunnel
Roosevelt Island
Northern Blvd
Greenpoint Ave
McGuinness Blvd
Grand St
Broadway
Flushing Ave
English Kills
East River Park
Sara D Roosevelt Park
McCarren Park
1 mi
1 km
©2008 Google - Map data ©2008 Tele Atlas - Terms of Use

widgets to the margins

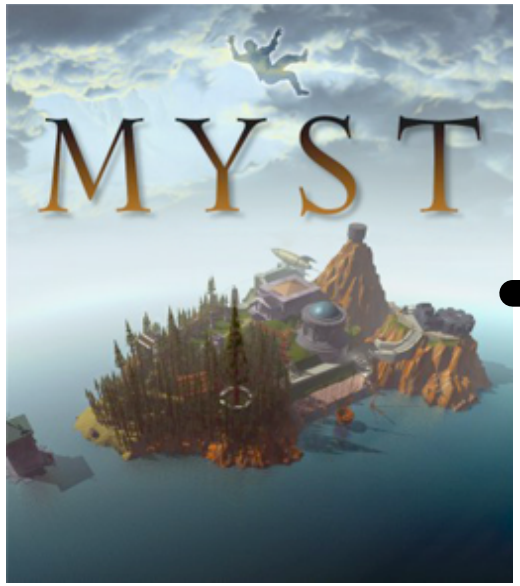
data front and center

widgets to the margins

positional awareness



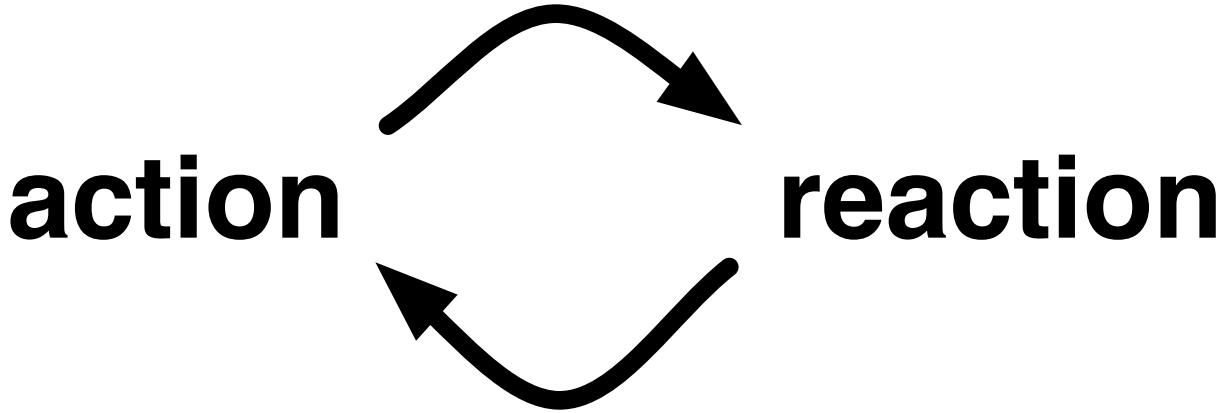
positional awareness



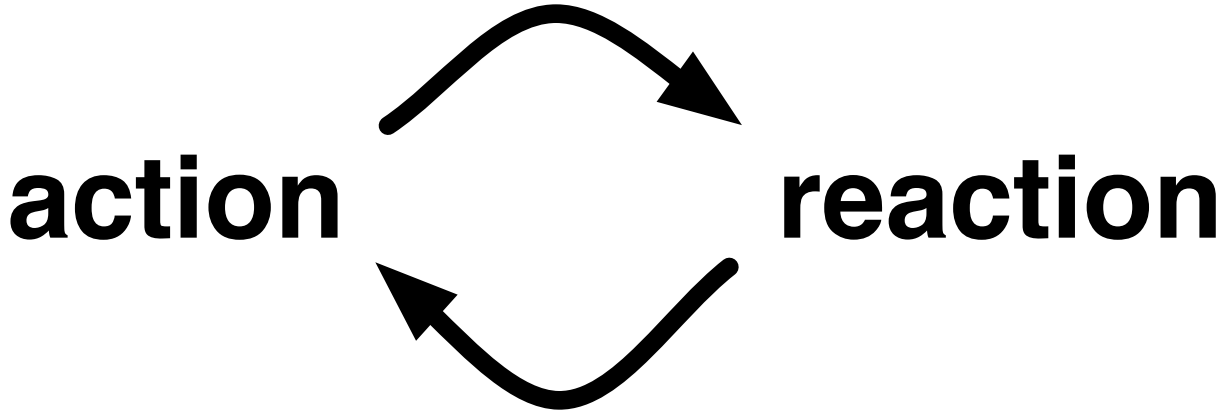
transitions

animations

fluidity

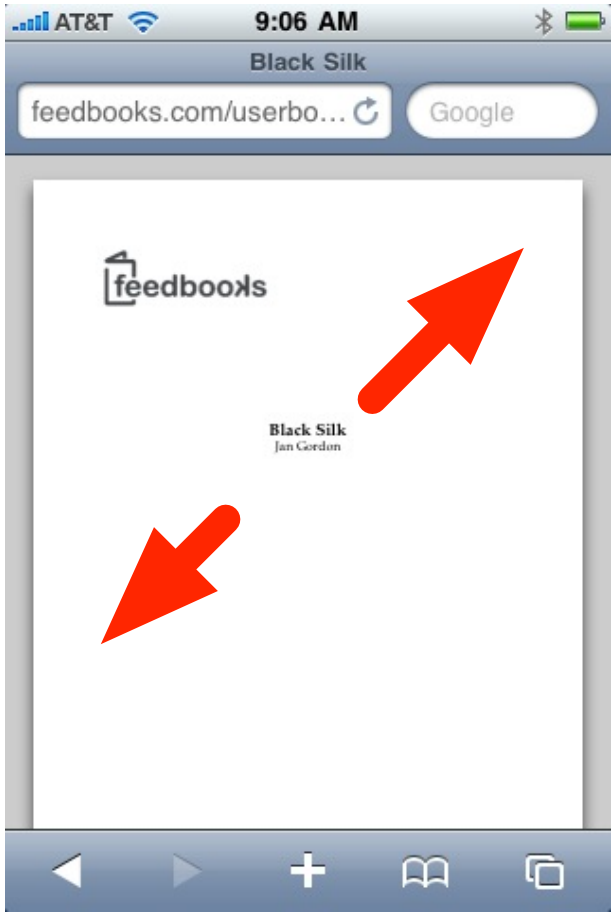


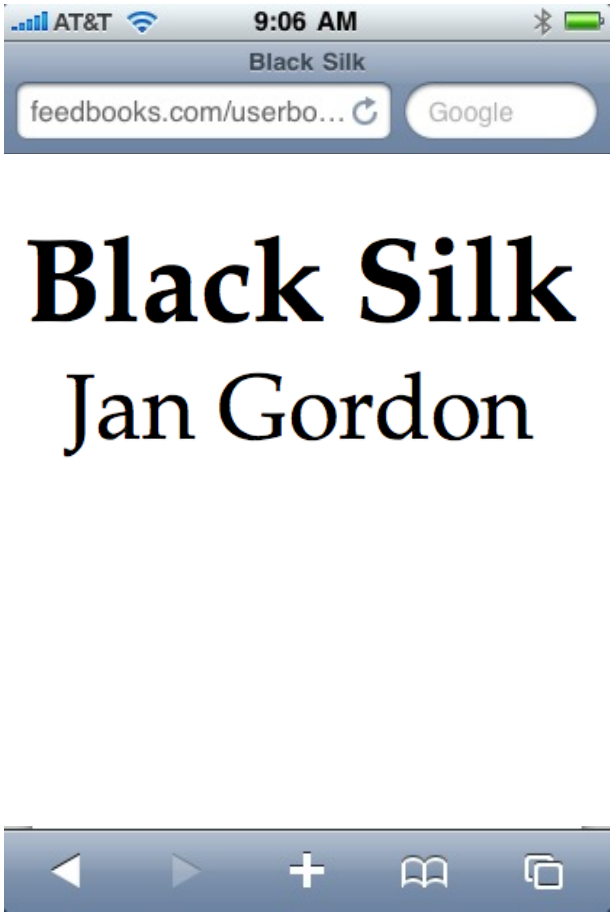
fluidity



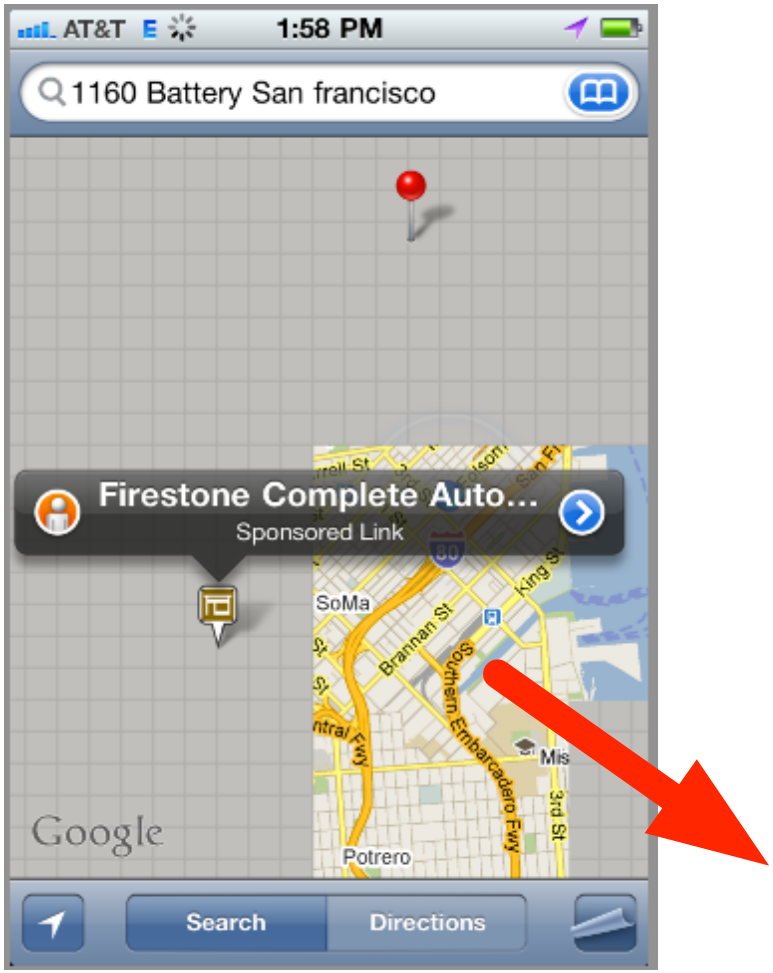
< 100ms

maintaining immersion





maintaining immersion



no spinners
no progress bars
no loading screens
just **drive**

can we do this for UCSC?

motivation

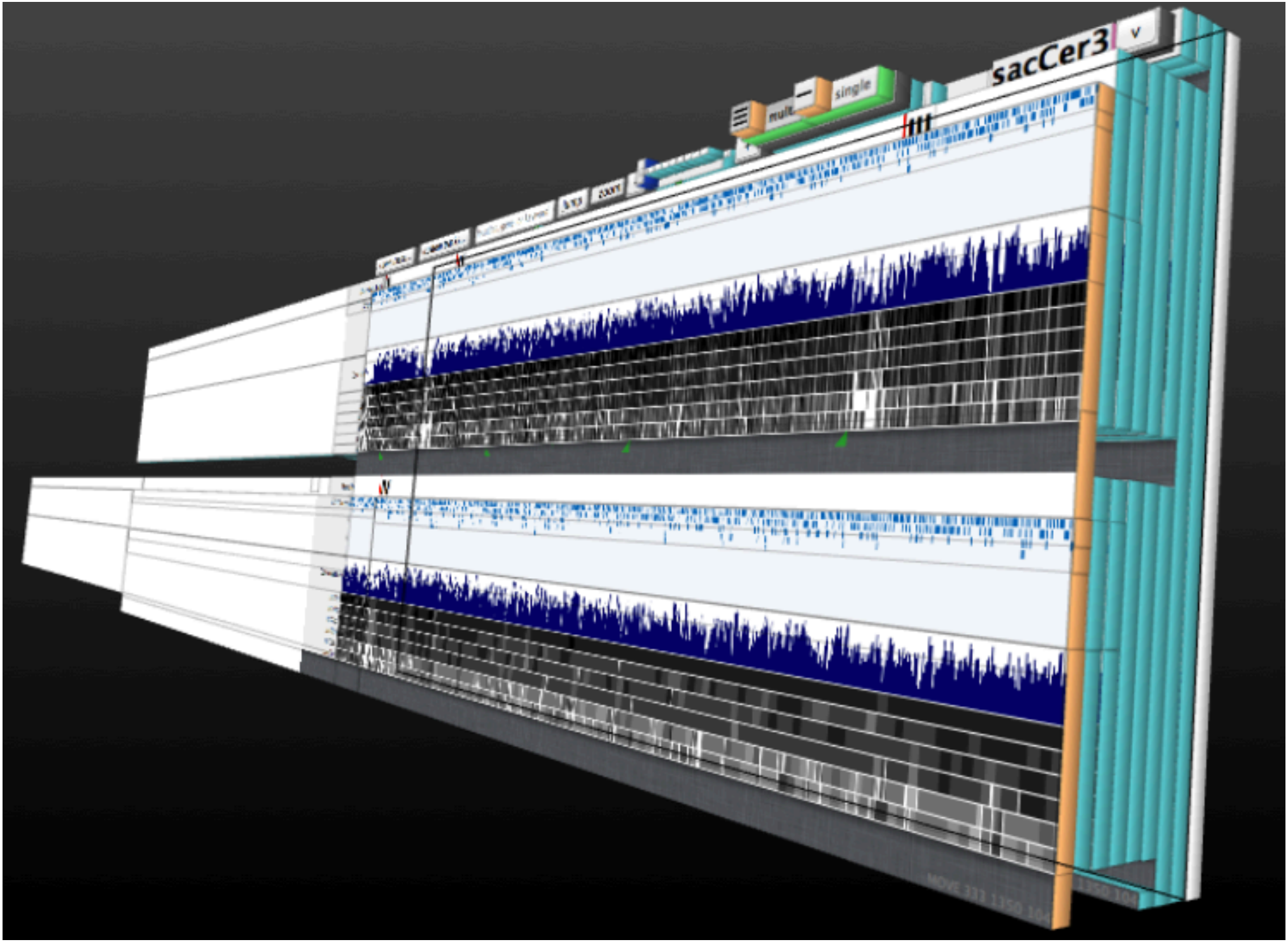
live demo

how it works

motivation

live demo

how it works

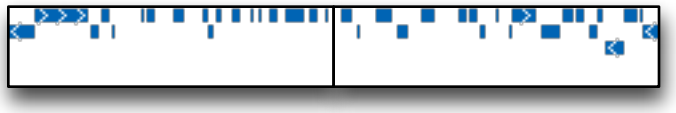

sacCer3
multi
single

tiling technique

bp / px

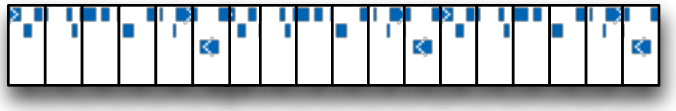
...

1.0e+3

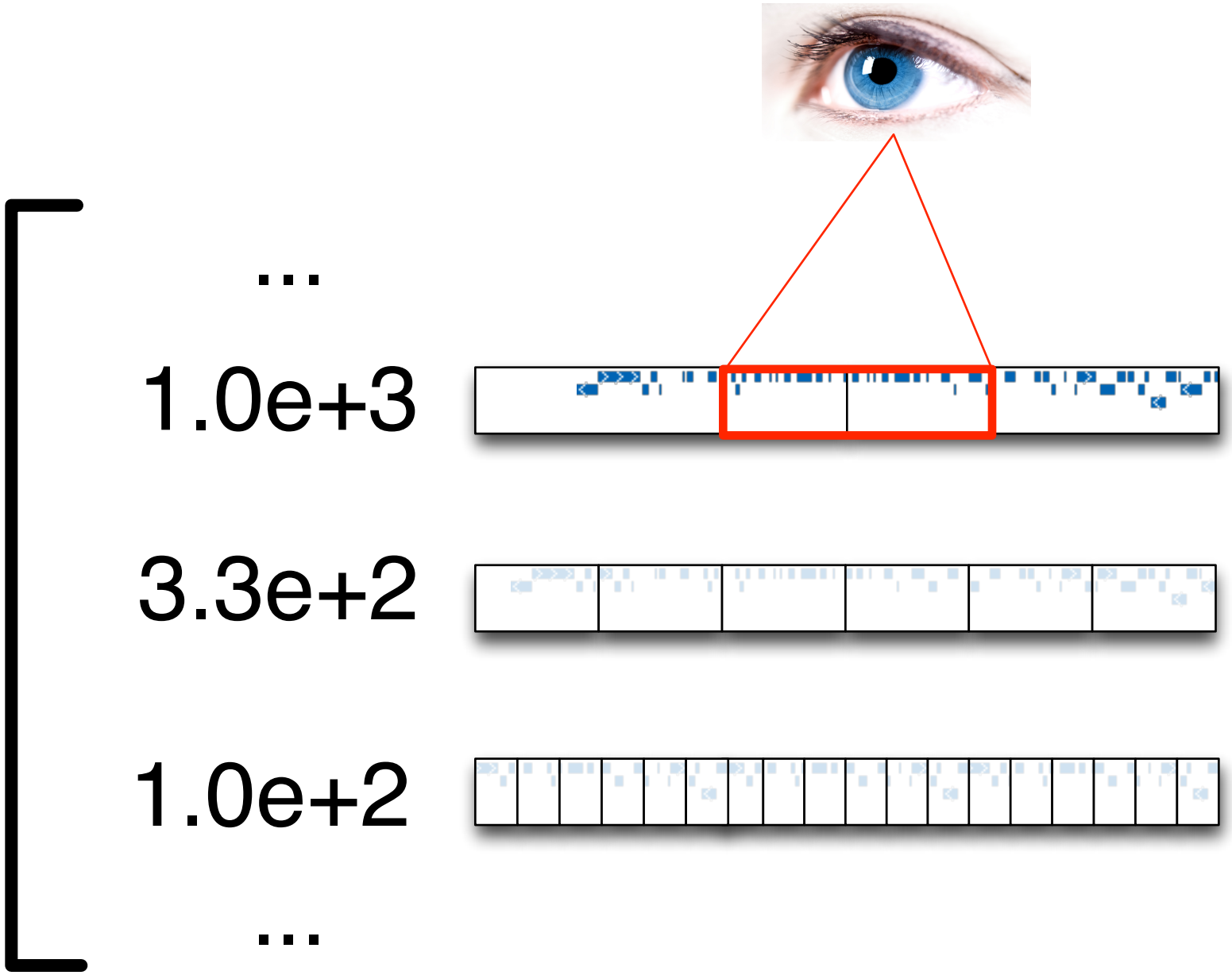


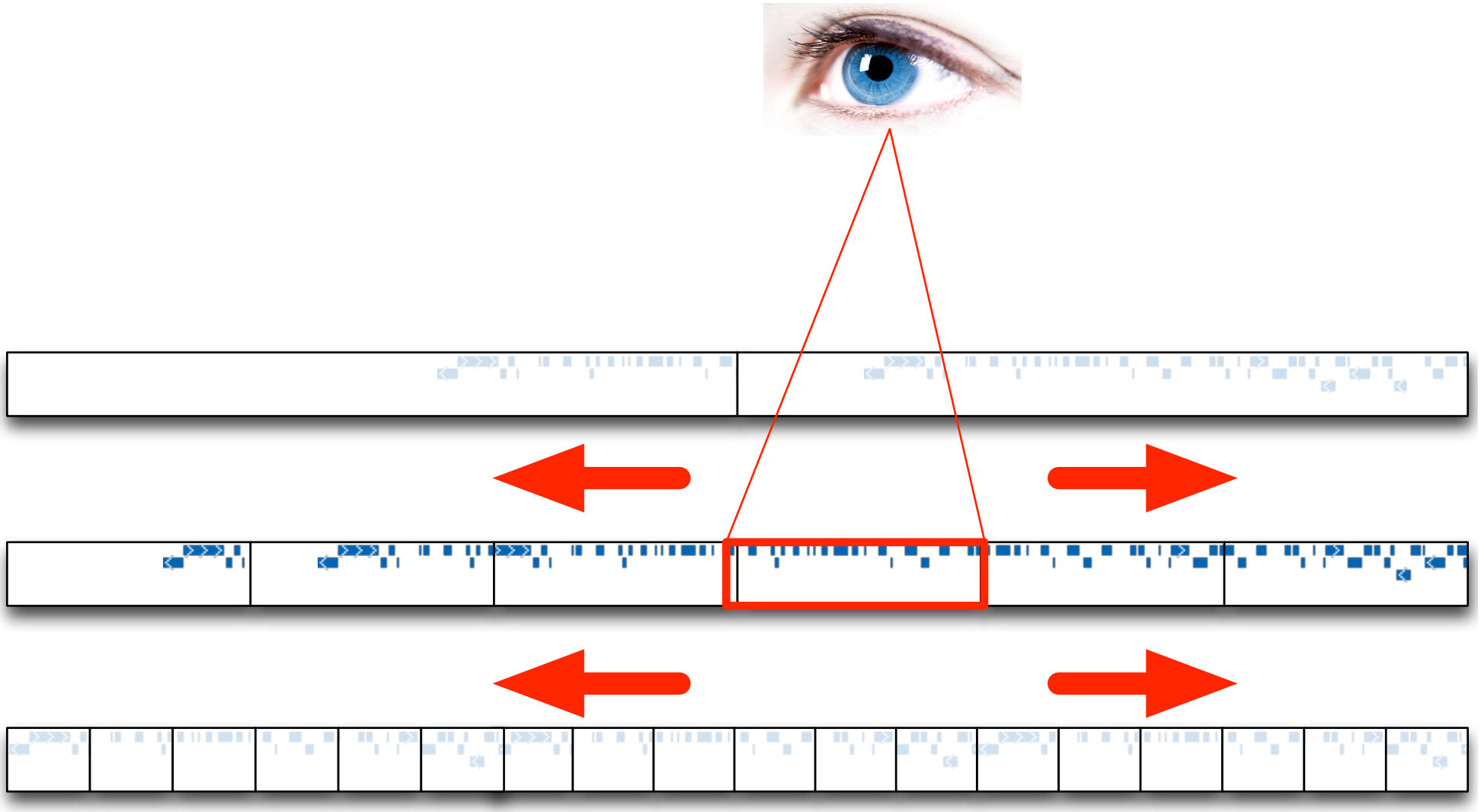
3.3e+2

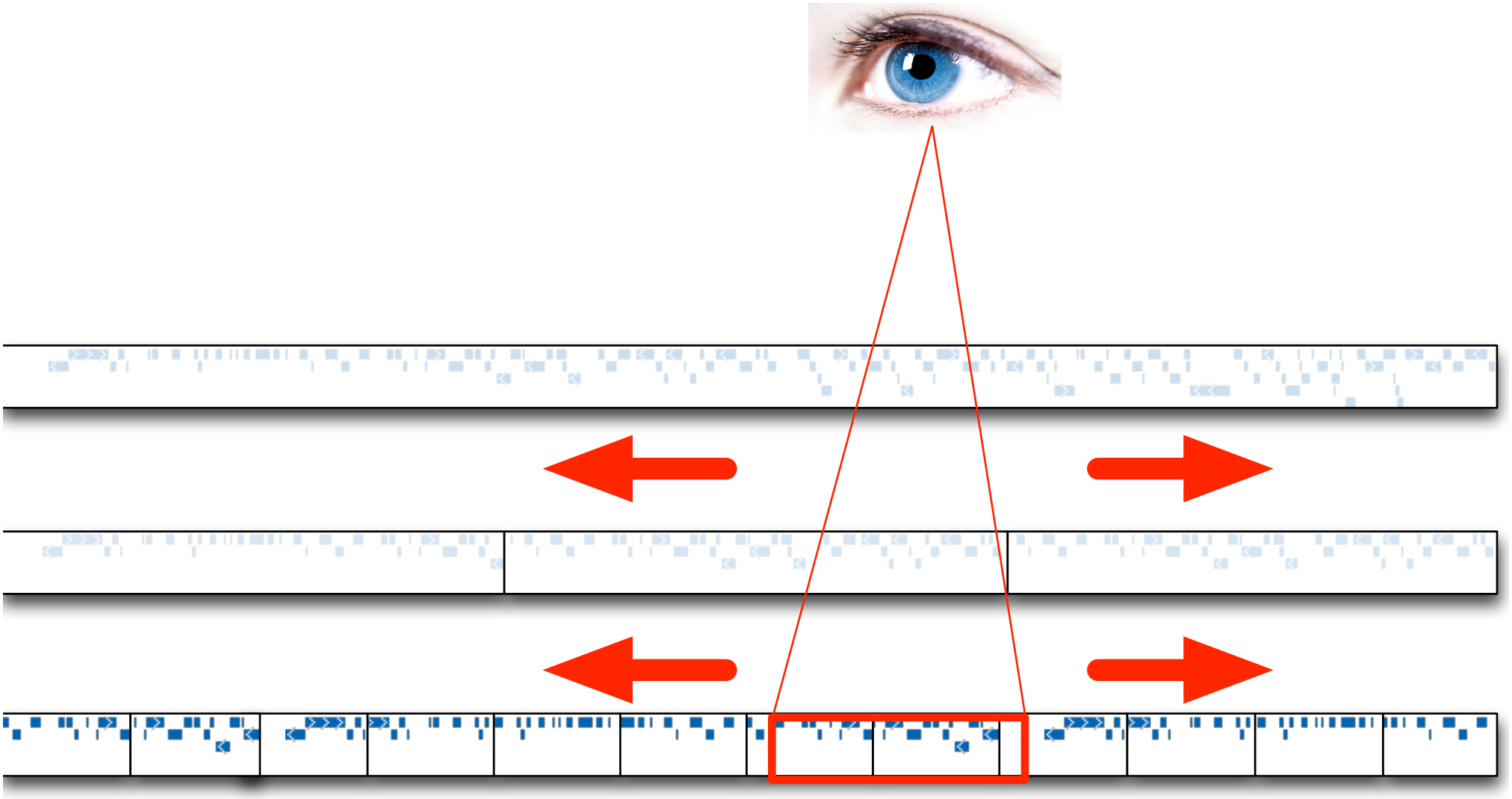
1.0e+2



...


...
1.0e+3
3.3e+2
1.0e+2
...



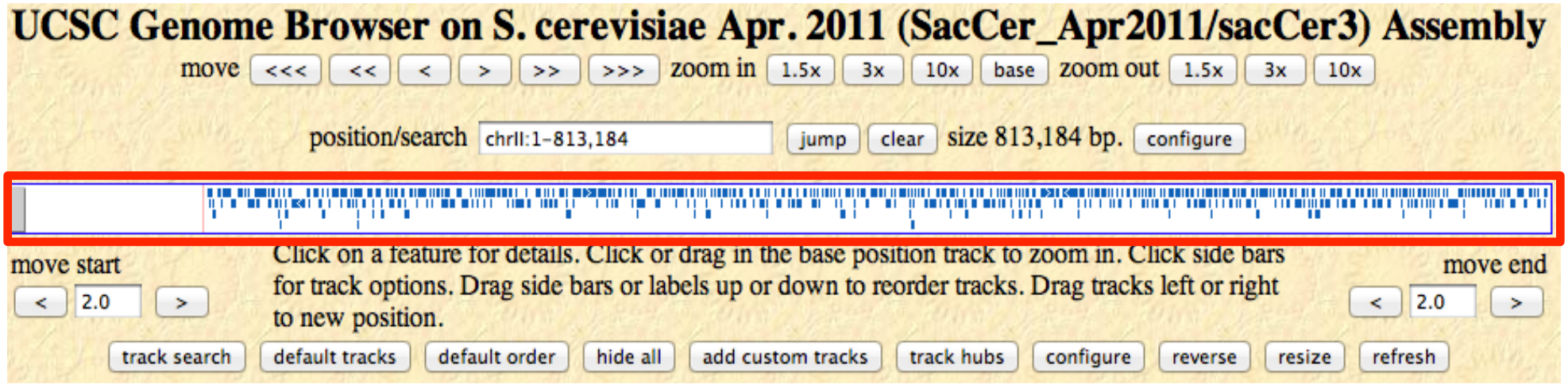


generating tiles

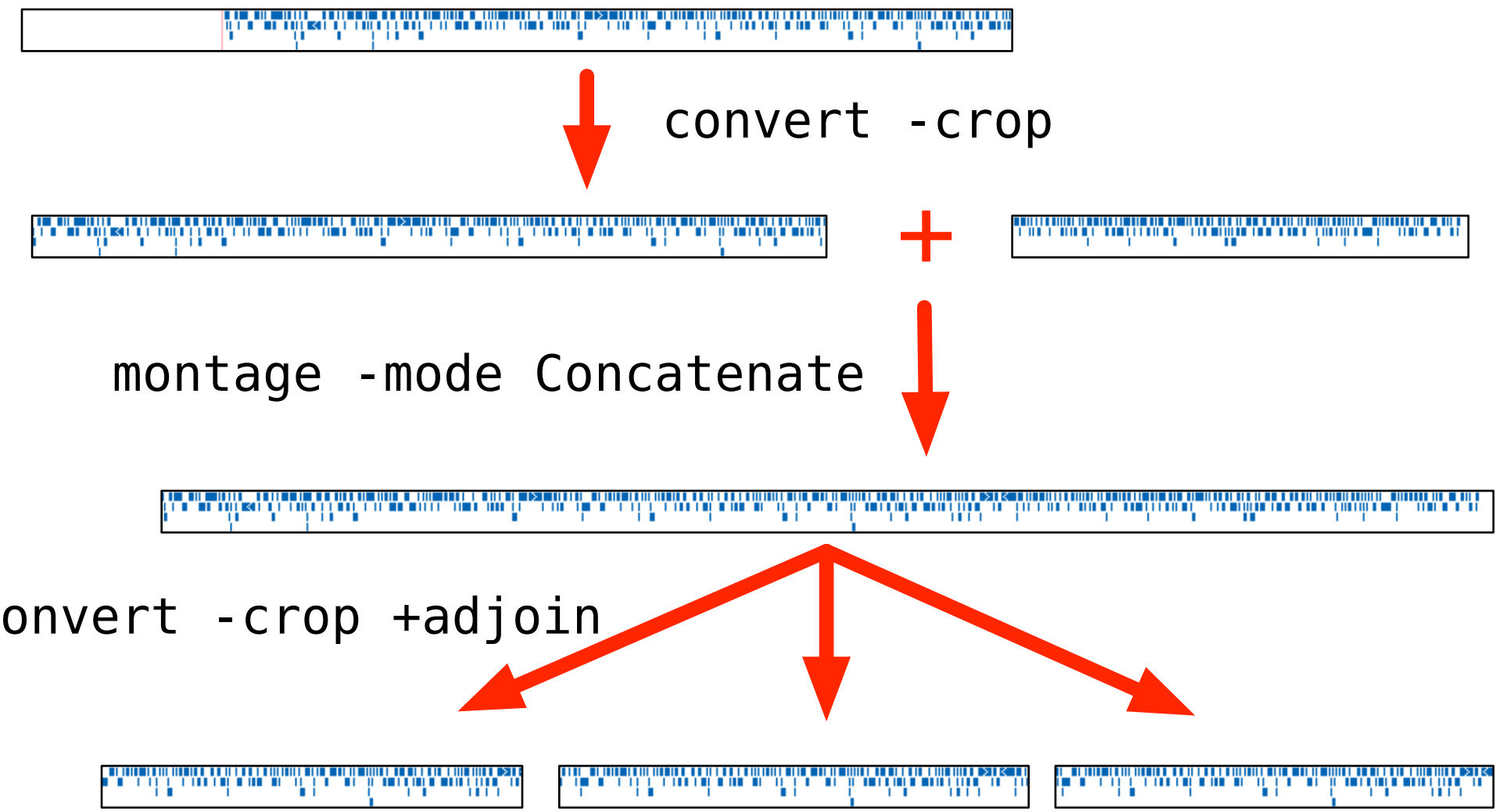
```
#!/usr/bin/env ruby
require 'rubygems'
require 'yaml'
require 'open-uri'
require 'nokogiri'
require 'tempfile'

class UCSCClient
  # ...
  def get_track_piece(track, chr, start, fin, bppp,
                      size='dense')
    base_uri = URI.parse(@ucsc_config['baseUrl'])
    uri = base_uri.clone
    opts = {}
  #...
```

nokogiri

```
doc = Nokogiri::HTML(uri.open)
nk = doc.xpath("//img[starts-with(@src,
                  '../trash/hgt/hgt_genome_')]")

temp_file = InterimFile.new(['ucsc','.png'], 'tmp/')
system("curl", "-s", (base_uri
  + nk.first['src']).to_s, "-o", temp_file.path)
```



imagemagick



tile "database"

```
/Volumes/HDD2$ find sacCer3
sacCer3
sacCer3/blastHg18KG
sacCer3/blastHg18KG/1.00e+00_dense
sacCer3/blastHg18KG/1.00e+00_dense/0000
sacCer3/blastHg18KG/1.00e+00_dense/0000/000001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/001001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/002001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/003001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/004001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/005001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/006001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/007001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/008001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/009001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/010001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/011001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/012001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/013001.png
sacCer3/blastHg18KG/1.00e+00_dense/0000/014001.png
...
```

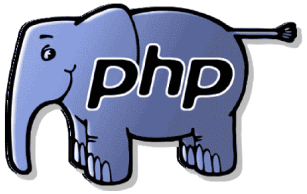
genome config

```
bppps:
    - 2.9e+5
    - 1.0e+5
    - 3.3e+4
    - 1.0e+4
    - 3.3e+3
    - 1.0e+3
    - 3.3e+2
    - 1.0e+2
    - 3.3e+1
    - 1.0e+1
tile_every: 1000
bppp_limits:
    ideogram: [3093, 1.0e+9]
    track: [0.1, 2.9e+5]
    ideograms_above: 1.1e+4
    nts_below: [1, 0.1]
    bppp_numbers_below: [3.3e+4, 1.0e+4]
chr_order: [chr1, chr2, chr3, chr4, chr5, chr6, chr7, chr8, chr9,
    chr10, chr11, chr12, chr13, chr14, chr15, chr16, chr17, chr18,
    chr19, chr20, chr21, chr22, chrX, chrY]
chr_lengths:
    chr1: 247249719
    chr2: 242951149
# ...
```

single page HTML5 app

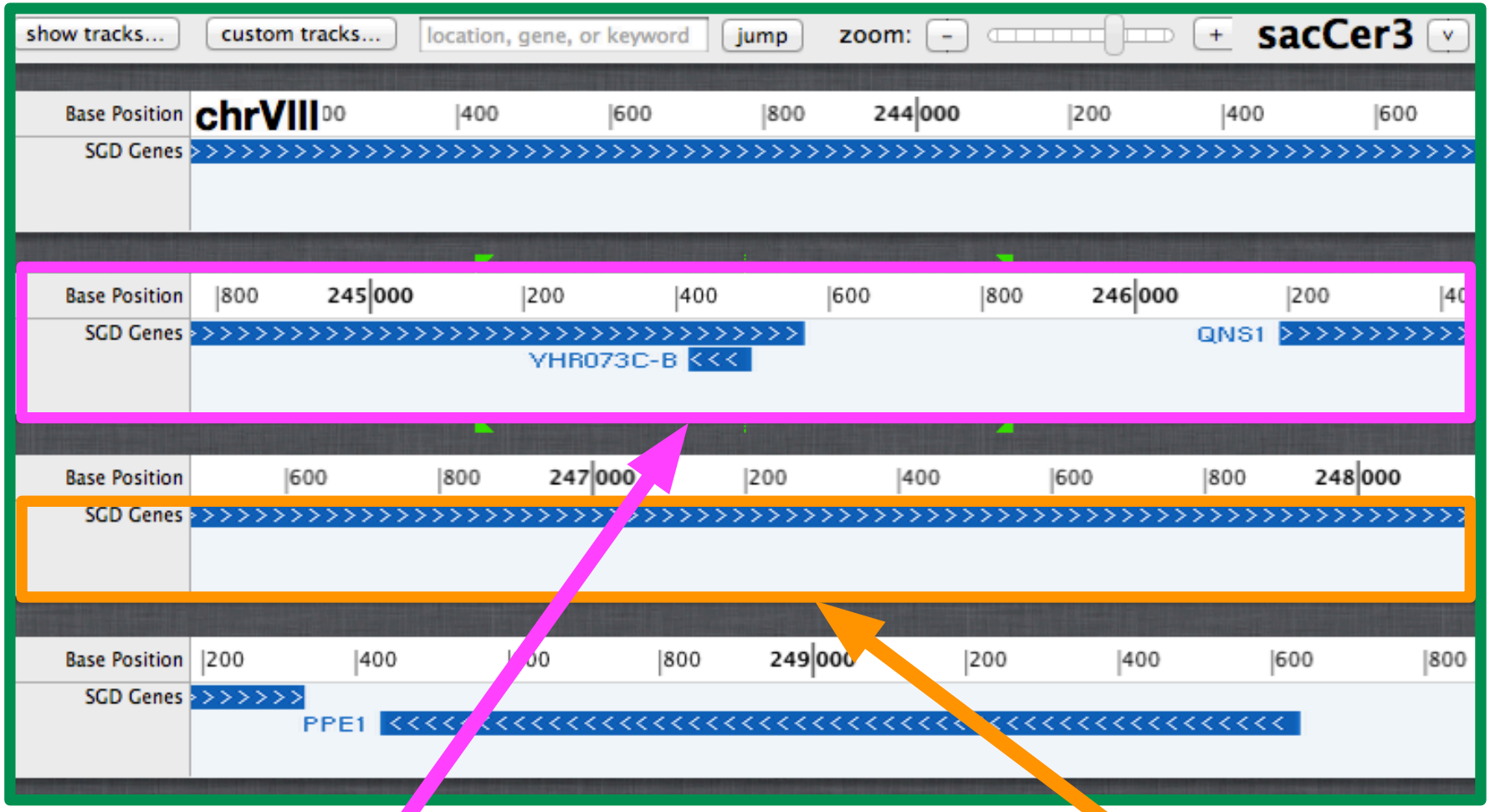
a **lot** of fancypants JavaScript

with a sprinkle of



widget hierarchy

$.ui.genobrowser



$.ui.genoline

$.ui.genotrack

why not use gmaps or OpenLayers API?

it's been done (XMap, Gen. Projector)

optimal for 2D, not 1D, navigation

locked into the limitations of the API

bothersome to "translate" coordinates

keeping high fps

Minimize DOM operations.

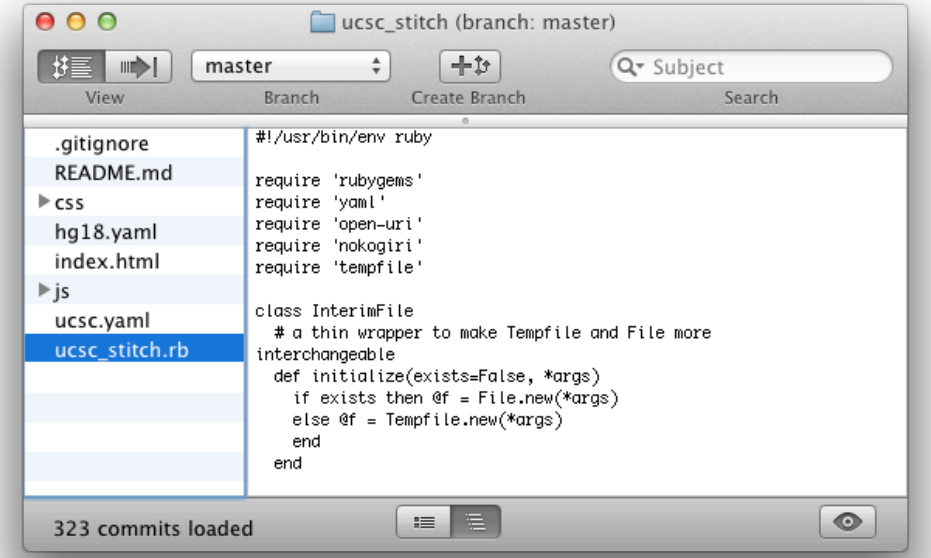
Minimize DOM operations!

Minimize # of DOM elements

Use `<canvas>` whenever possible

Webkit Inspector: profile, refactor

version 1



1. write YAML config for genome
2. run Ruby script, generate tiles
3. start webserver
4. open index.html in browser

problem 1

scraping over the internet is slow

(and rude)

solution

install UCSC locally

Procedure for Creating a Mirror Site for the UCSC Genome Browser

The following procedure provides you with step-by-step instructions to incrementally create a mirror of the UCSC Genome Browser. You may choose to set up either a full mirror browser or a partial one, depending on your disk space and needs. See also: Minimal Browser Installation instructions on genomewiki. Additionally, these blog posts may be helpful on setting up a mirror site on CentOS and Ubuntu.

3 weeks later...

(I keed, I keed…)

pro solution

run the CGI binaries directly

```
Dir.mktmpdir do |dir|
  Dir.chdir(dir) do
    resp = `#{@ucsc_config['cgi_bin_dir']}/hgTracks '#{uri.query}'`
    # get rid of HTTP headers before passing to Nokogiri
    doc = Nokogiri.parse(resp(/(.*\n)*\n\n/, ''))
    yield doc, false
  end
end
```

(saves overhead of Apache and HTTP)

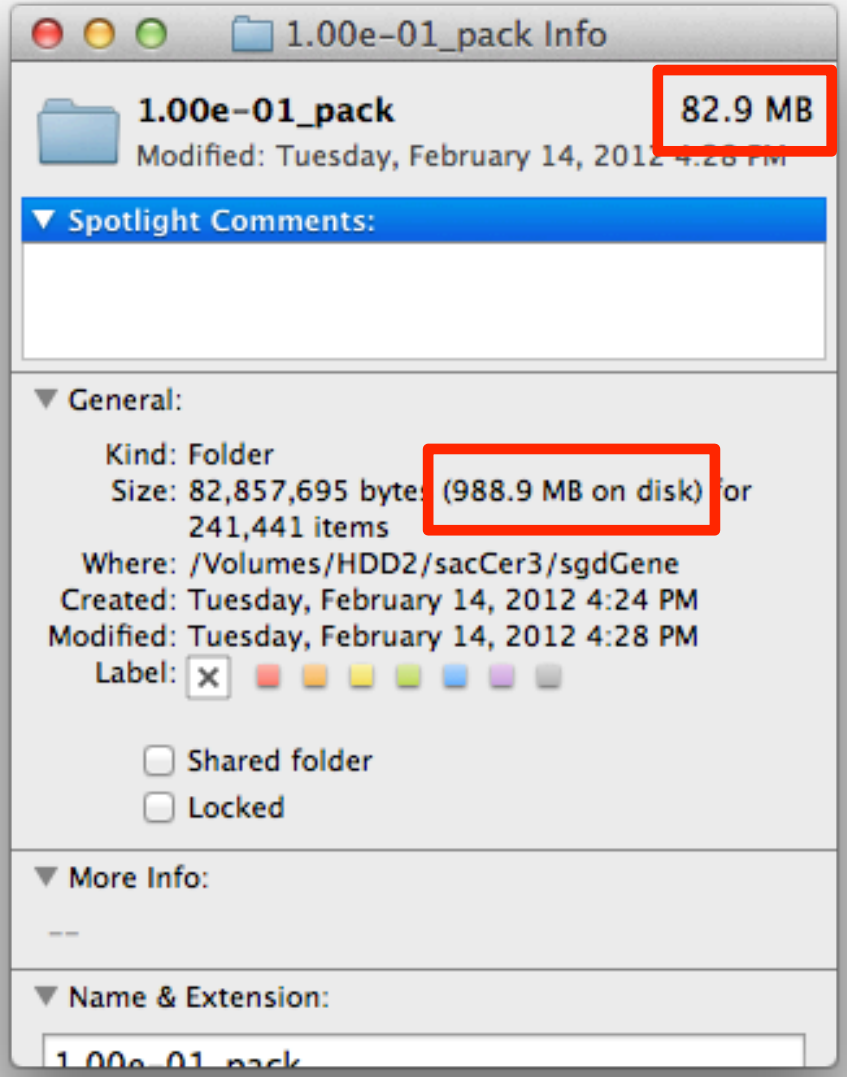
problem 2

we are wasting

tons

of disk space

(and the filesystem is getting slow)



lots of <4kB files

=

lots of partial blocks

=

wasted HDD

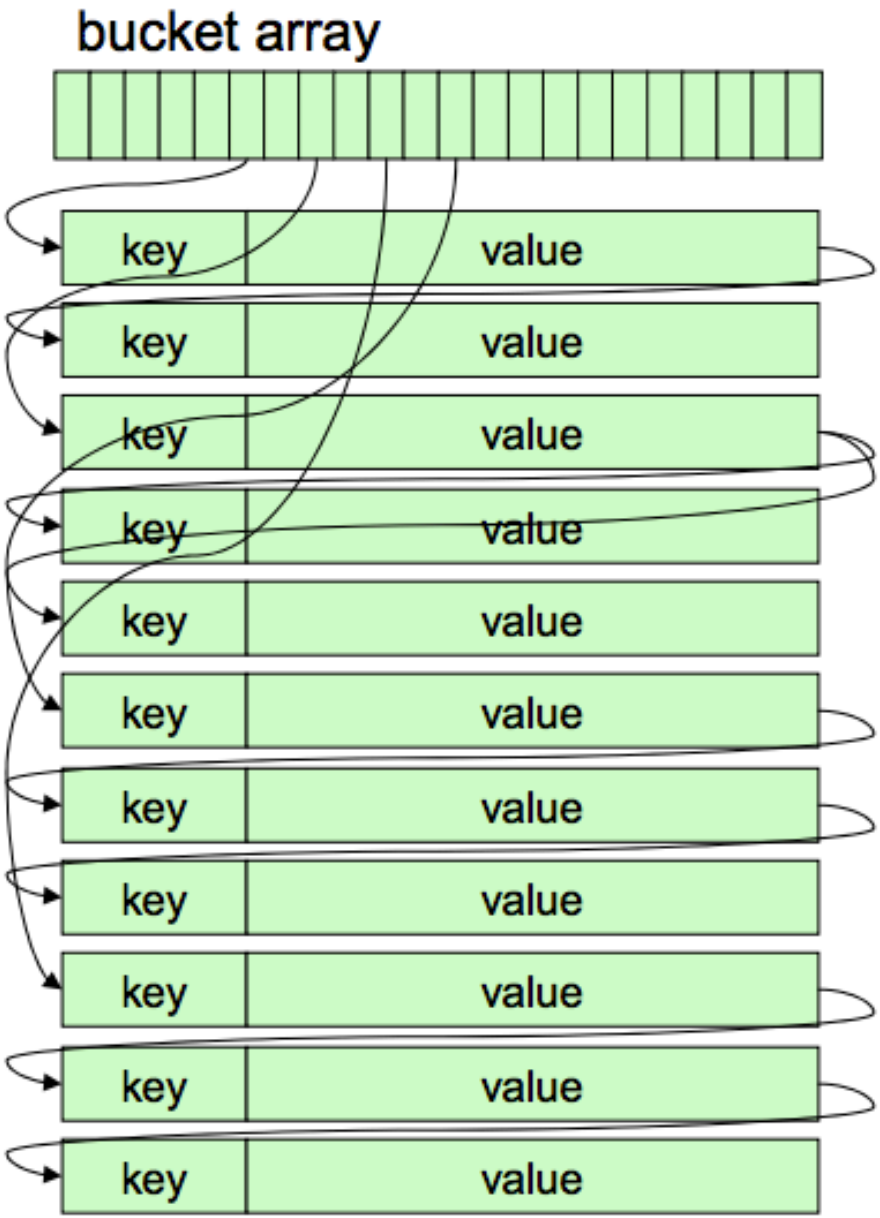
solution

use an on-disk hashtable

ooooh.

Tokyo Cabinet 8192PiB

look ma noSQL



why tokyo?

- based on DBM
- O(1) hashing & lookup
- ~2 seeks per read
- **fast** and **simple**
  - 2.5M inserts/sec locally
  - 100K qps over a network

problem 3

running the ruby script
is single-threaded.
tile stitching is slow.

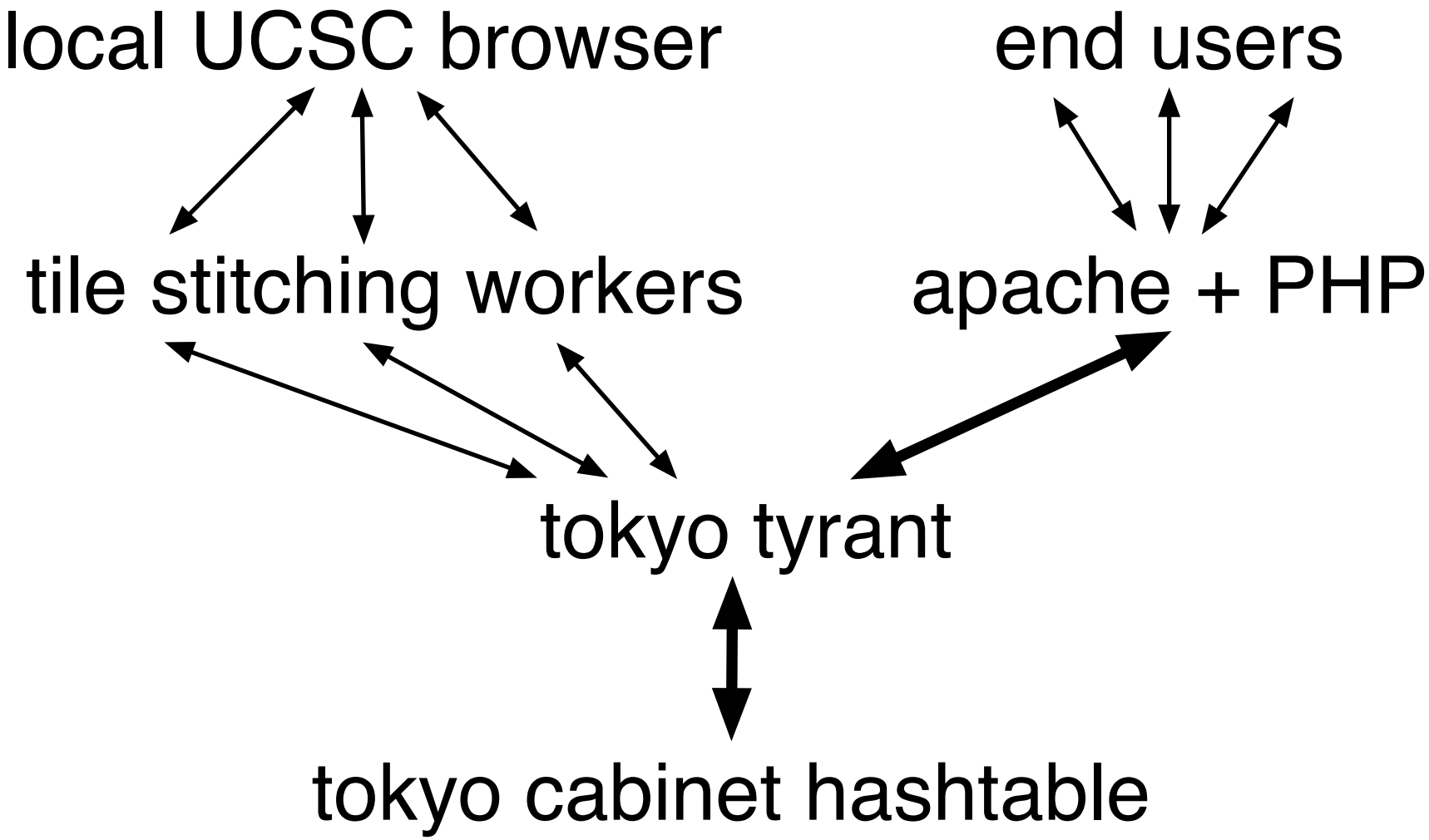
solution

1. refactor as rake task
2. parallelize:
   - make lockfiles w/ `File.flock`
   - multiple processes can divvy up tracks and generate tiles
3. run on the cluster

rake: Ruby make

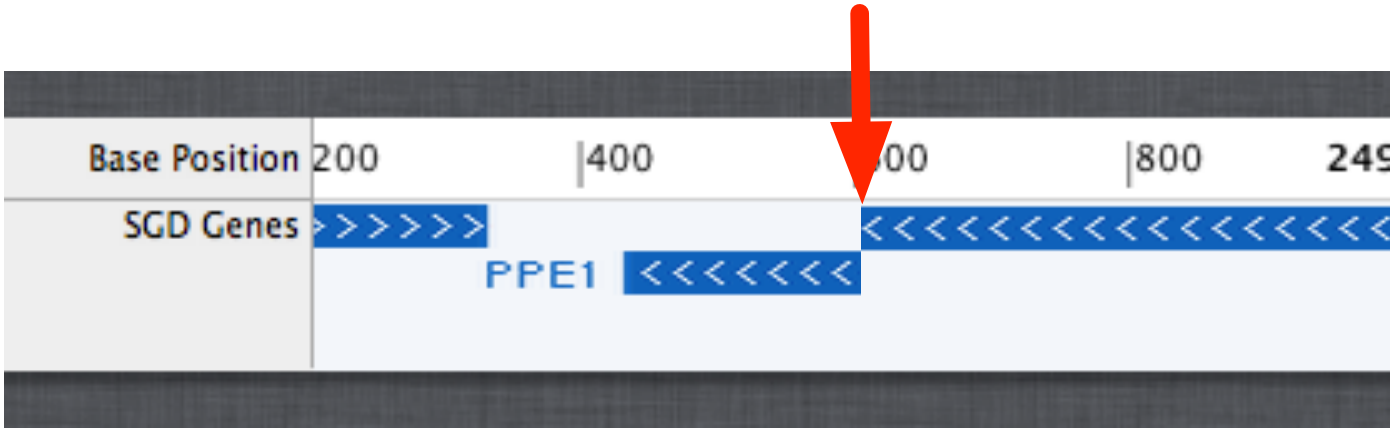
```
~/src/ucsc_stitch$ rake -T
...
rake check
  # Checks that all requirements for UCSCin are in place

rake config[genome]
  # Interactively create a base YAML configuration file for a...

rake json[genome,skip_tiles]
  # Rebuilds the JSON file that holds a genome's configuration for...

rake json_clean[genome]
  # Deletes the JSON file that holds a genome's configuration for...

rake stat_tiles[genome,exhaustive]
  # Check the status of tracks for a genome

rake tch[genome]
  # Creates/updates a Tokyo Cabinet hashtable from an existing...

rake tiles[genome,exhaustive,workers]
  # Create tiles for a genome (optionally using multiple workers)
```

final architecture



problem 4

tiles can have "seams" where UCSC rendered the same feature on different rows


Base Position
200
400
800
SGD Genes
PPE1

some grepping later

```
~/src/kent/src/hg/lib$ grep -A4 -B4 5000 trackLayout.c
#ifdef LOWELAB
    if (tl->picWidth > 60000)
      tl->picWidth = 60000;
#else
    if (tl->picWidth > 5000)
      tl->picWidth = 5000;
#endif
    if (tl->picWidth < 320)
        tl->picWidth = 320;
    }
```

hmm...

solution

bump up the image width limit
from 5000 px
to 100000 px

patch + recompile

```
$ diff -ru src/hg/lib/trackLayout.c src/hg/lib/trackLayout.c
--- src/hg/lib/trackLayout.c 2012-02-21 13:01:54.000000000 -0500
+++ src/hg/lib/trackLayout.c 2012-02-27 16:35:14.000000000 -0500
@@ -20,9 +18,14 @@
     if (tl->picWidth > 60000)
       tl->picWidth = 60000;
 #else
+#ifdef ROTHLAB
+    if (tl->picWidth > 100000)
+      tl->picWidth = 100000;
+#else
     if (tl->picWidth > 5000)
       tl->picWidth = 5000;
 #endif
+#endif
```

problem 5

ImageMagick is slow and is hogging memory

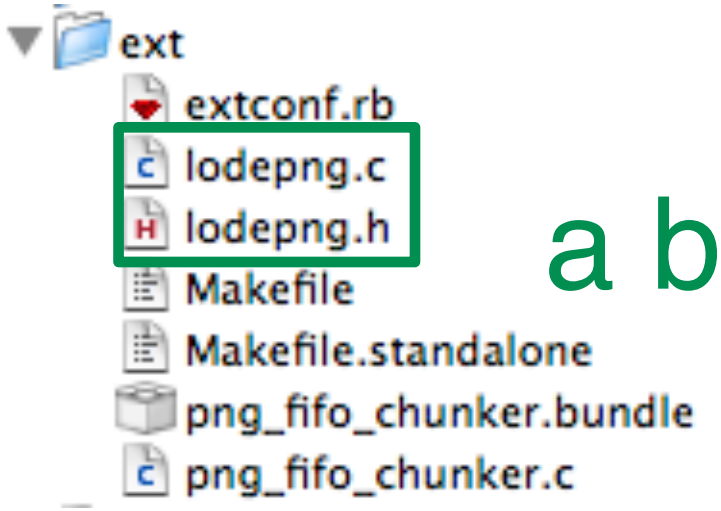
RSS of workers > real memory ➔ swapping ➔ slow death.

solution

build a ruby extension in C
for image processing
in the inner loop

ruby makes this easy

```
~/src/ucsc_stitch/ext$ cat extconf.rb
# Loads mkmf which is used to make makefiles for Ruby extensions
require 'mkmf'

$CFLAGS << ' -ggdb -O0' if ARGV.size > 0 && ARGV[0] == 'debug'

# Give it a name
extension_name = 'png_fifo_chunker'

# The destination
dir_config(extension_name)

# Do the work
create_makefile(extension_name)

~/src/ucsc_stitch/ext$ ruby extconf.rb && make config && make
```



lodepng

a barebones PNG library

http://lodev.org/lodepng/

```
~/src/ucsc_stitch/ext$ cat png_fifo_chunker.c
#include "lodepng.h"
#include "ruby.h"

// ...

VALUE PNGFIFO_chunk_split(int argc, VALUE *args, VALUE self) {
  // ...
}

// The initialization method for this module
void Init_png_fifo_chunker() {
  Module = rb_define_module("PNGFIFO");
  rb_define_method(Module, "chunk_split", PNGFIFO_chunk_split, -1);
}
```

current stats

Can render hg18
8 default tracks, all densities
@ 1bppp
using 48 workers
in about 3 days.

Database size: 80GB

final problem!

custom tracks

...

we will never be able to pre-render them fast enough

solution

use some HTML5 magic
to render them browser-side
right next to the standard tracks.

live demo

reading the files

For local files:

- HTML5 File API

For remote files:

- AJAX proxy

Pass to web workers for parsing

problem: JS blocks UI updates
solution: web workers

what are they?

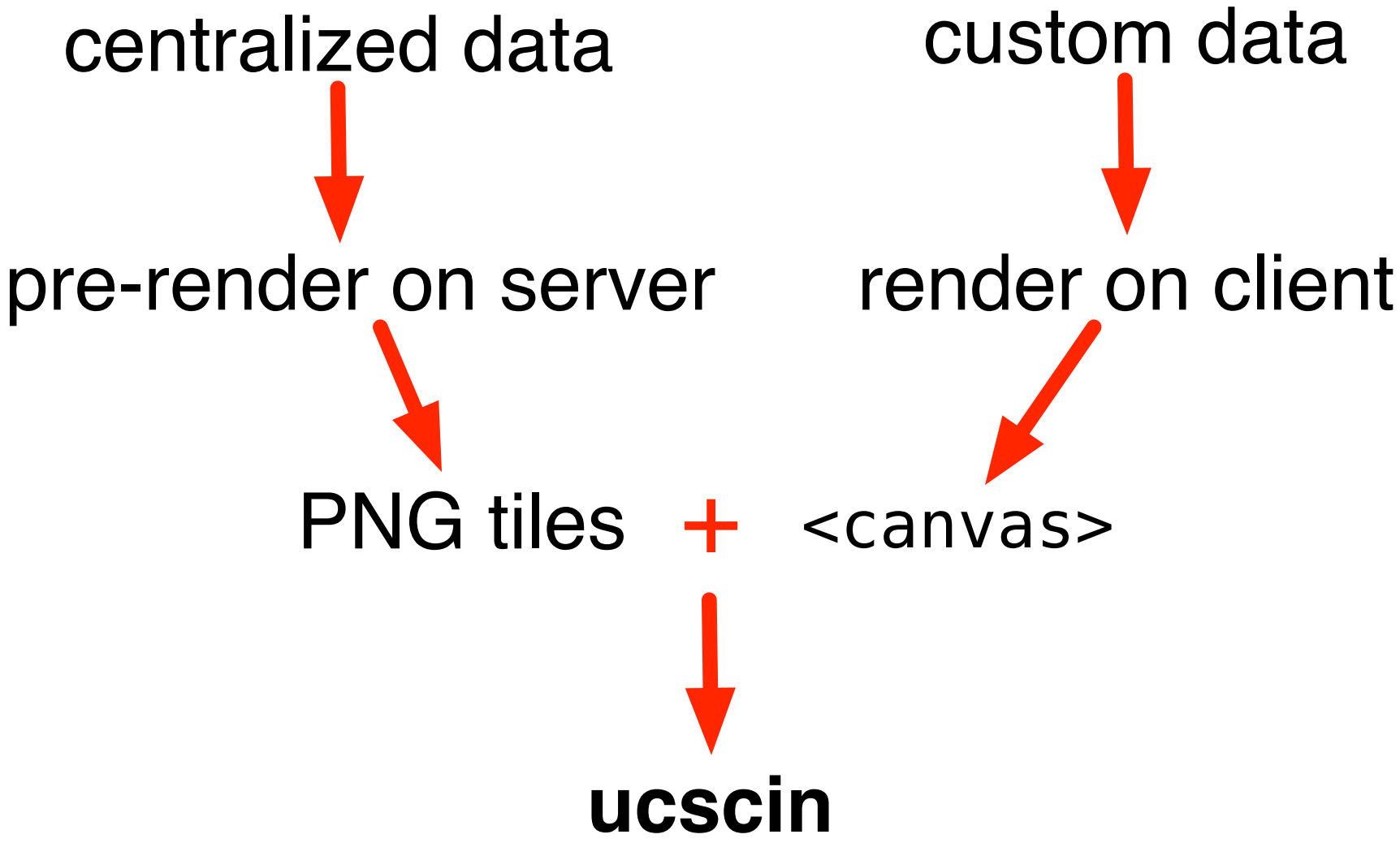
- Full-fledged JS interpreters
- Run in background processes
- Communicate via message passing
- Cannot access DOM directly

```
global.addEventListener('message', function(e) {
  var data = e.data,
    callback = function(r) {
      global.postMessage({
        id: data.id,
        ret: JSON.stringify(r || null)
      });
    },
    ret;

  try {
    ret = CustomTrackWorker[data.op].apply(CustomTrackWorker,
      data.args.concat(callback));
  } catch (err) {
    // handle errors
  }

  if (!_.isUndefined(ret)) { callback(ret); }
});
```

rendering

- Drawn in <canvas> elements
- Can do:
  - **BED** and **bigBed** (exons only)
  - **WIG** and **bigWig**
  - **VCFTabix**
- Should be easy to add more
- **big*** formats: best performance

division of labor



comparison with

- JBrowse
- AnnoJ
- ABrowse
- GBrowse
- NCBI
- Ensembl

one more thing...

(still quite alpha)

TODO

- release on github
- better help & error handling
- documentation on custom trax
- more track formats
- more URL params + history
- (bug squashing)
- …please suggest more!

thank you!

in particular,
my advisor Fritz Roth
the entire Roth laboratory
and all of you for inviting me.

building howto

1. install prereqs
   - ruby, rake, a few gems
   - ImageMagick, curl, libxml2
2. clone repo
3. `$ rake check`

   `$ rake`
4. will generate config + tiles