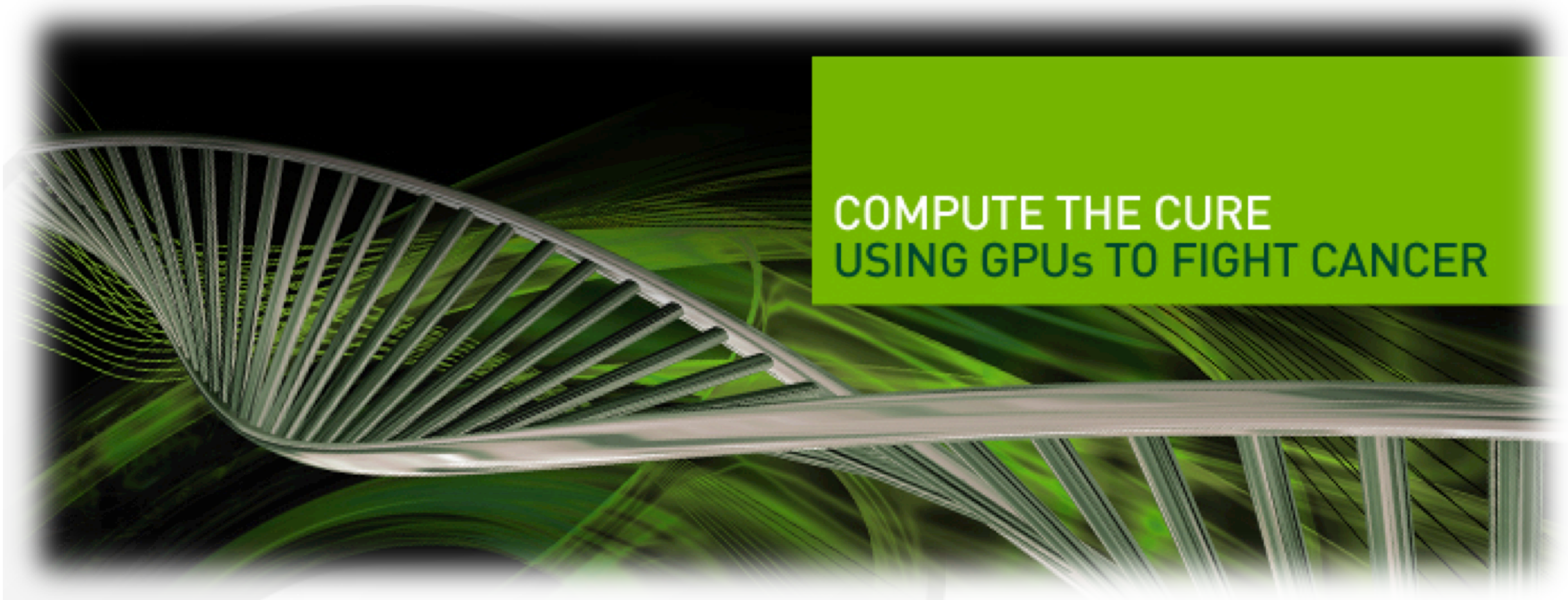


Towards Computing the Cure for Cancer

Wu Feng, PhD
Department of Computer Science
Department of Electrical & Computer Engineering
Heshan Lin, PhD
Department of Computer Science

VirginiaTech
Invent the Future

SyNeRG
synergy.cs.vt.edu

COMPUTE THE CURE

Facts about Cancer

- How frequent does a person die from cancer in the U.S.?
  - *Once every MINUTE*
- How many *new* cases of cancer diagnosed worldwide in 2007?
  - *More than 12 MILLION (12,000,000)*
- How many died from cancer in 2007?
  - *7.6 MILLION,* making it *the* leading cause of death worldwide
- What are the conservative projections for 2050?
  - *New Cases: More than 27 MILLION*
  - *Deaths: 17.6 MILLION* if our ability to prevent, diagnose and treat cancer does not improve

VirginiaTech Invent the Future

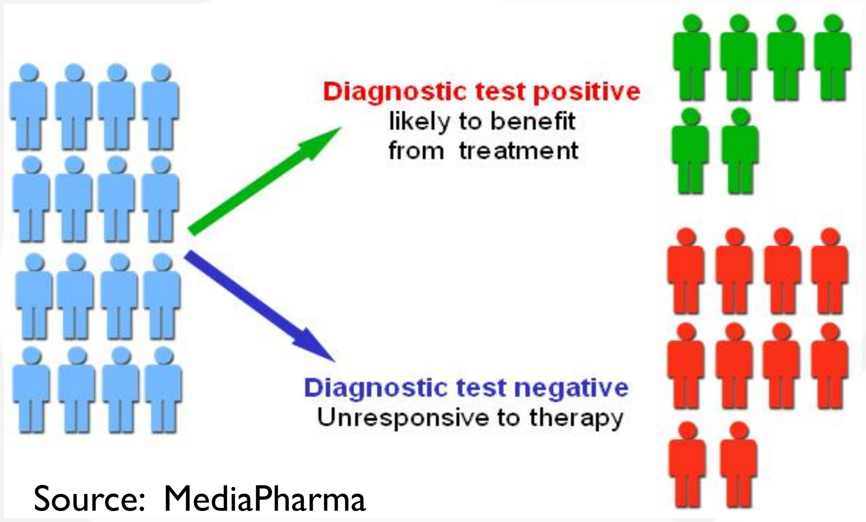
Sources: ICGC, TCGA, WHO

SyNeRG synergy.cs.vt.edu

Goals of Cancer Genome Research

- Identify changes in the genomes of tumors

  … that drive cancer progression
- Identify new targets for therapy
- Select drugs based on the genomics of the tumor



Source: MediaPharma

The Ultimate Goal

The right treatment

… at the right dose

… for the right patient

… at the right time

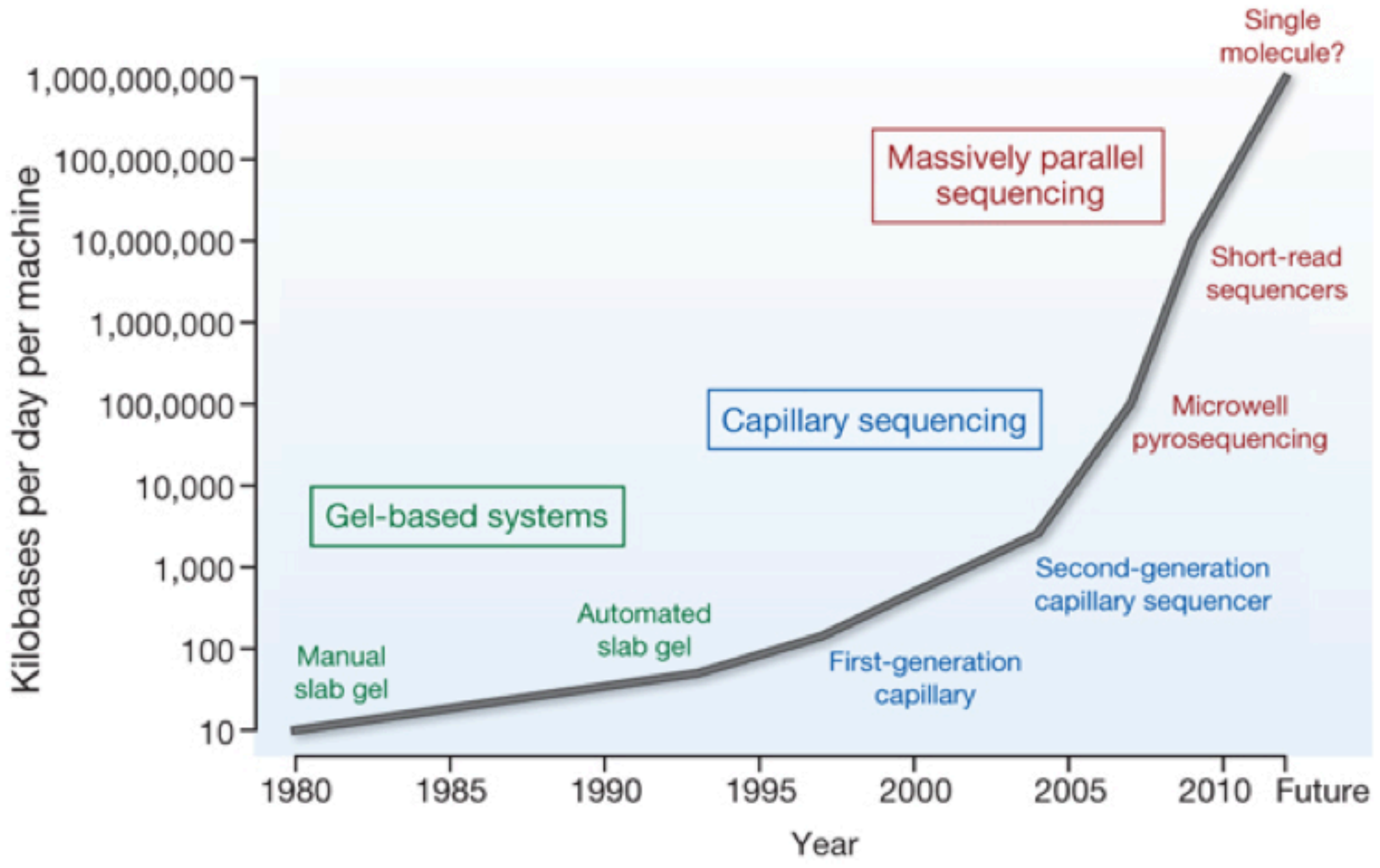
… for the right outcome

Source: ICGC

Large-Scale Cancer Genome Studies

- Johns Hopkins U. (Wood et al., *Science*, Oct. 2007)
  - More than 18,000 genes analyzed for mutations
  - 11 breast and 11 colon tumors
- Welcome Trust Sanger Institute (Greenman et al., *Science*, Mar. 2007)
  - 518 genes analyzed for mutations
  - 210 tumors of various types
- The Cancer Genome Atlas (Collins & Barker, *Sci. Am.*, Mar. 2007)
  - Multiple technologies to map genetic changes of 20 cancers
- International Cancer Genome Consortium
  - Identify genomic, transcriptomic, and epigenomic changes in 50 tumor types

Sequencing Throughput



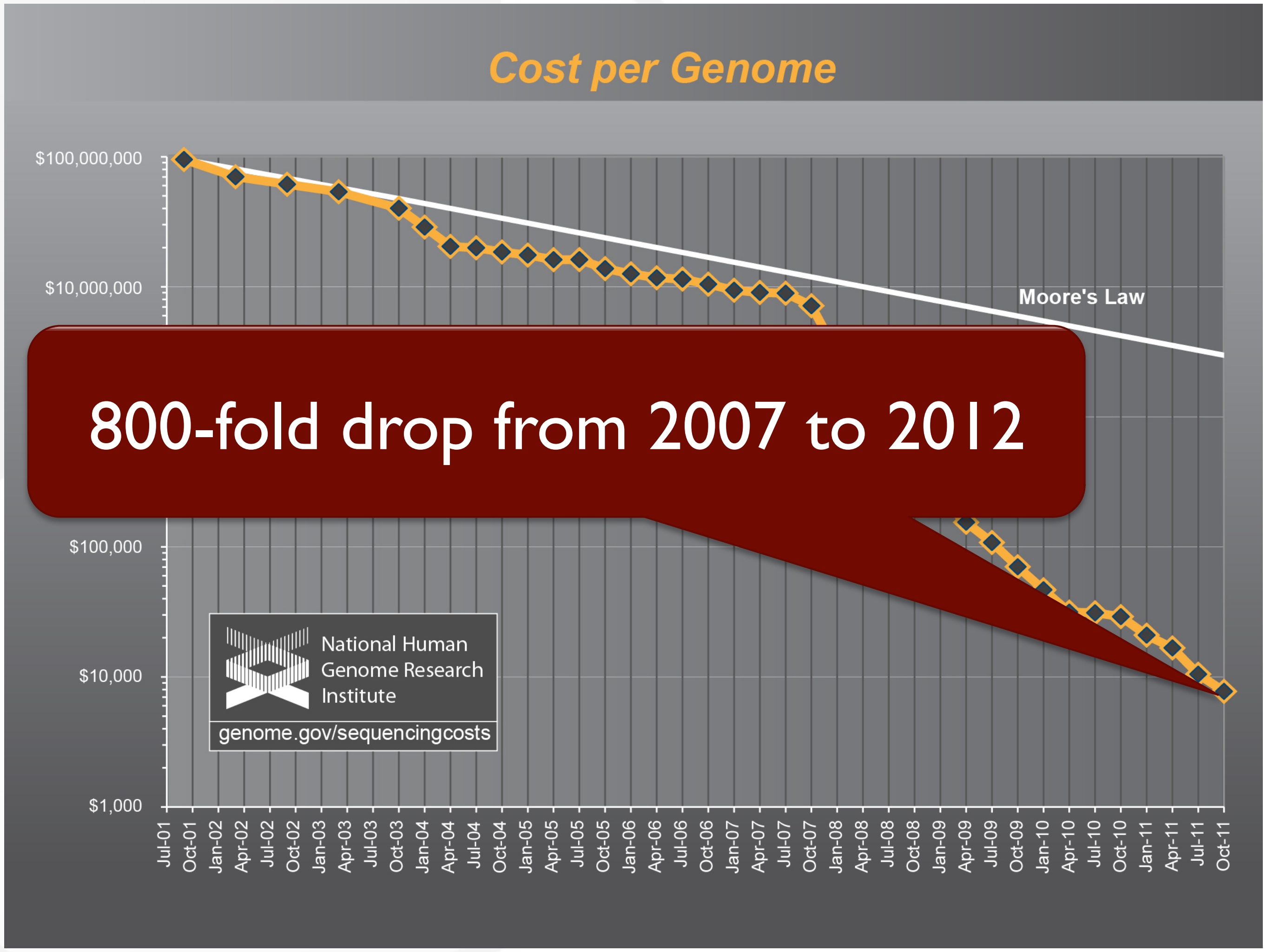
MR Stratton *et al.* *Nature* **458**, 719-724 (2009)

Invent the Future

synergy.cs.vt.edu

COMPUTE THE CURE

Cost of DNA Sequencing



800-fold drop from 2007 to 2012

VirginiaTech *Invent the Future*

SyNeRG synergy.cs.vt.edu

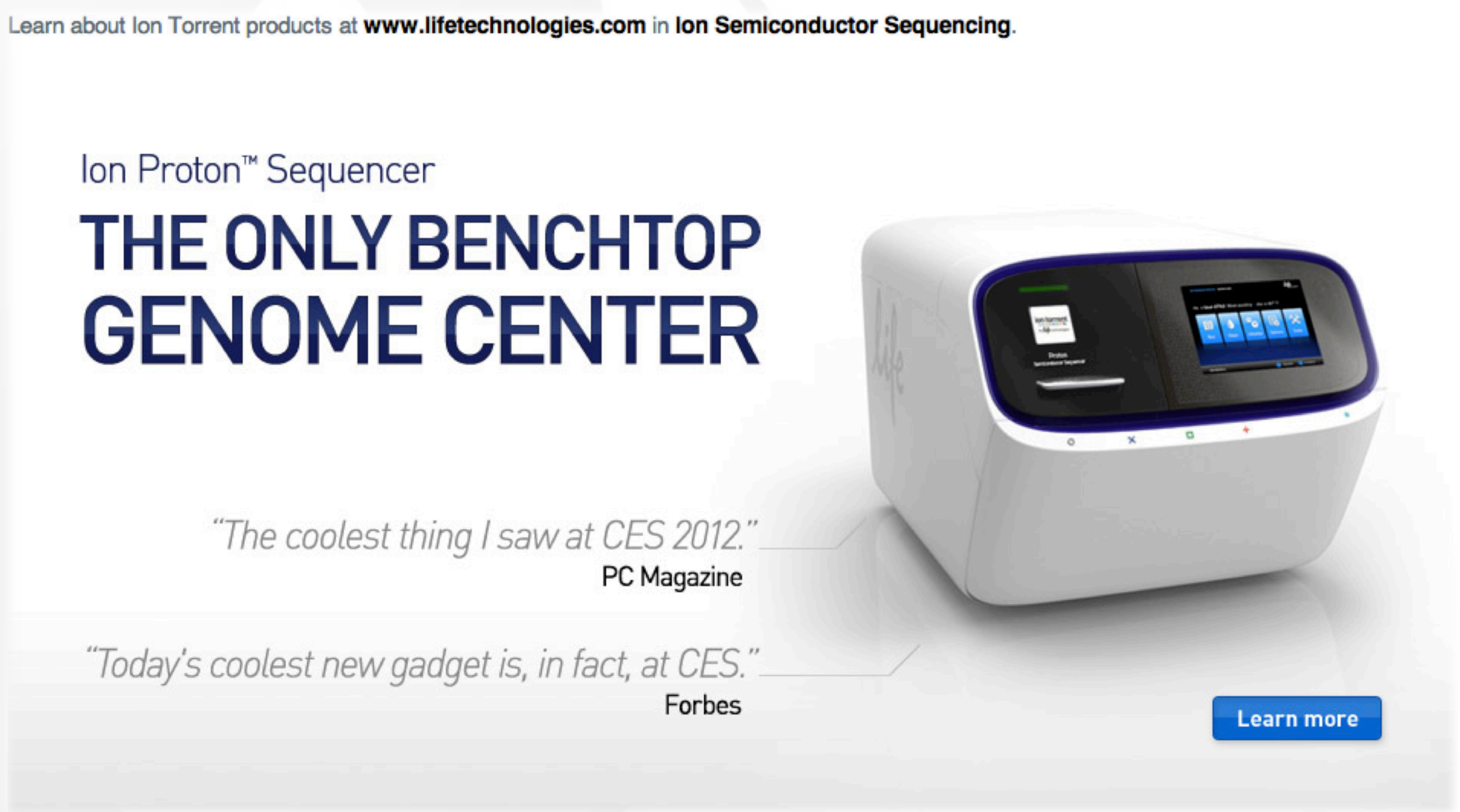
Next-gen sequencing (NGS) presents many opportunities to understanding cancer genome changes

Challenges of Next-Generation Sequencing (NGS) for Cancer

- Efficiently store and *analyze* massive amounts of DNA data

COMPUTE THE CURE

Personalizing NGS … *Not the Analysis*


Learn about Ion Torrent products at **www.lifetechnologies.com** in **Ion Semiconductor Sequencing**.

Ion Proton™ Sequencer

THE ONLY BENCHTOP GENOME CENTER

"The coolest thing I saw at CES 2012."
PC Magazine

"Today's coolest new gadget is, in fact, at CES."
Forbes

VirginiaTech
Invent the Future

SyNeRG
synergy.cs.vt.edu

Towards Personalizing NGS Analysis

Sampling of our work

Accelerating Protein Sequence Search in a Heterogeneous Computing System

Shucai Xiao*, Heshan Lin†, and Wu-chun Feng*†

cuBLASTP

On the Robust Mapping of Dynamic Programming onto a Graphics Processing Unit

GPU-RMAP: Accelerating Short-Read Mapping on Graphics Processors

A Maintainable Software Architecture for Fast and Modular Bioinformatics Sequence Search

Missing genes in the annotation of prokaryotic genomes

Andrew S Warren[1,2*], Jeremy Archuleta[2], Wu-chun Feng[2], João Carlos Setubal[1,2*]

Parallel Genomic Sequence-Search on a Massively Parallel System

Parallel Genomic Sequence-Searching on an Ad-Hoc Grid: Experiences, Lessons Learned, and Implications

A Pluggable Framework for Parallel Pairwise Sequence Search

Jeremy Archuleta, Wu-chun Feng, Eli Tilevich

Short-Read Mapping

- Bfast
- BioScope
- Bowtie/Bowtie2
- BWA
- CLC bio
- CloudBurst
- Eland/Eland2
- GenomeMapper
- GnuMap
- Karma
- MAQ
- MOM
- Mosaik
- MrFAST/ MrsFAST
- NovoAlign
- PASS
- PerM
- RazerS
- RMAP
- SSAHA2
- Segemehl
- SeqMap
- SHRiMP/SHRiMP2
- Slider/Slider II
- SOAP/SOAP2
- Srprism
- Stampy
- Vmatch
- ZOOM

… and so on

Pain Points for Cancer Biologist

- Time to Solution
  - Sequencing throughput >> compute throughput
  - Days to analyze (instead of hours or even minutes)
- Ease of Use
  - Steep learning curve to identify right tools, use tools, and integrate & compose tools

Key Unmet Need in NGS

"Lack of *user-friendly tools* to decipher the *large amount of data* generated by next-generation sequencing (NGS)."

Source: DeciBio, November 2011

Which bio tool do I use and how do I use it?

How do I integrate the use of tools from my toolbox?

COMPUTE THE CURE

Towards Computing the Cure for Cancer

http://www.computethecure.org/

- Empower scientists to fight cancer

  … through innovative parallel computing

- Foster a community

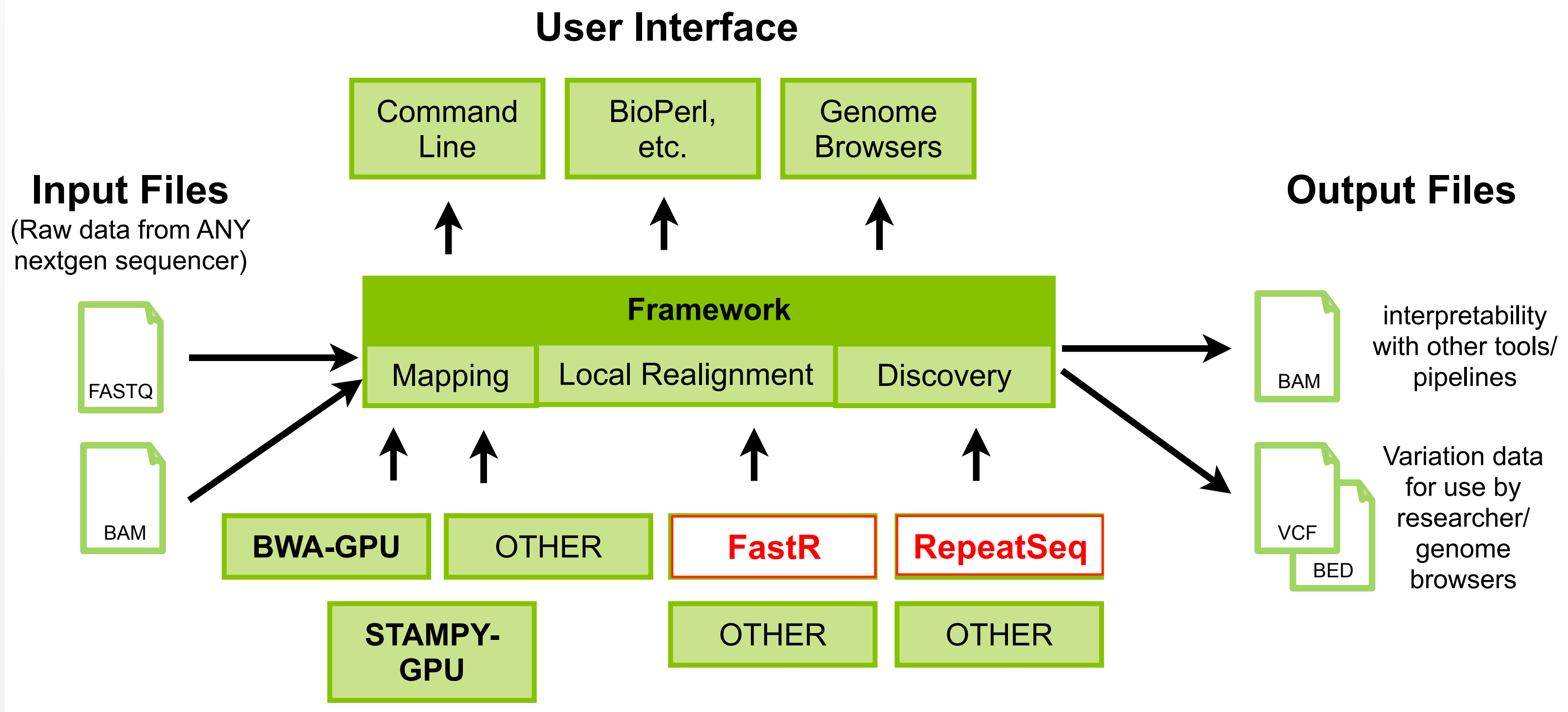
  … for developing accelerated bioinformatics tools

- Develop an easy-to-use genome analysis framework

  … to allow cancer biologists to focus on the science of cancer rather than on the *computer* science

A Framework for Genome Analysis

→ Open Genomics Engine (OpenGE)



COMPUTE THE CURE
Phase I

Improved Tools

Novel Tools

Source: NVIDIA Foundation & D. Mittelman
(Inspired by GATK @ Broad Institute)

synergy.cs.vt.edu

Overall Status of OpenGE

- Open-source software framework for cancer researchers to improve the productivity (i.e., speed and ease of use) with which to identify DNA mutations that lead to cancer.
- Sample OpenGE Workflows
  - BWA → GATK IndelRealigner → GATK Genotyper
  - BWA → FastR → Dindel
  - BWA → SAMtools
- Primary OpenGE Plug-Ins
  - Short-Read Mapping: BWA and (soon) CUSHAW
  - Local Realignment: ***FastR*** and GATK Realignment
  - Discovery: Dindel and ***RepeatSeq***

gpucomputing.net
Connect • Communicate • Collaborate

Teaser: *Beyond* OpenGE

GPU and the 13 Dwarfs

View | Forums

Welcome to the "GPU and the 13 Dwarfs" community.

- Dr. Wu Feng

Why? →

- Hardware design that keeps future applications in mind
- Basis for future applications? 13 computational dwarfs

Example: N-body

- Fermi
  - 400M interactions (200,000 bodies)
  - 1M particles/second
- Kepler
  - 789M interactions (280,875 bodies)
  - 10M particles/second → billions of years of simulation

Similar Idea for OpenGE

- Abstract common algorithmic components
- Provide a library of GPU-accelerated components for building high-performance analysis (plug-in) tools

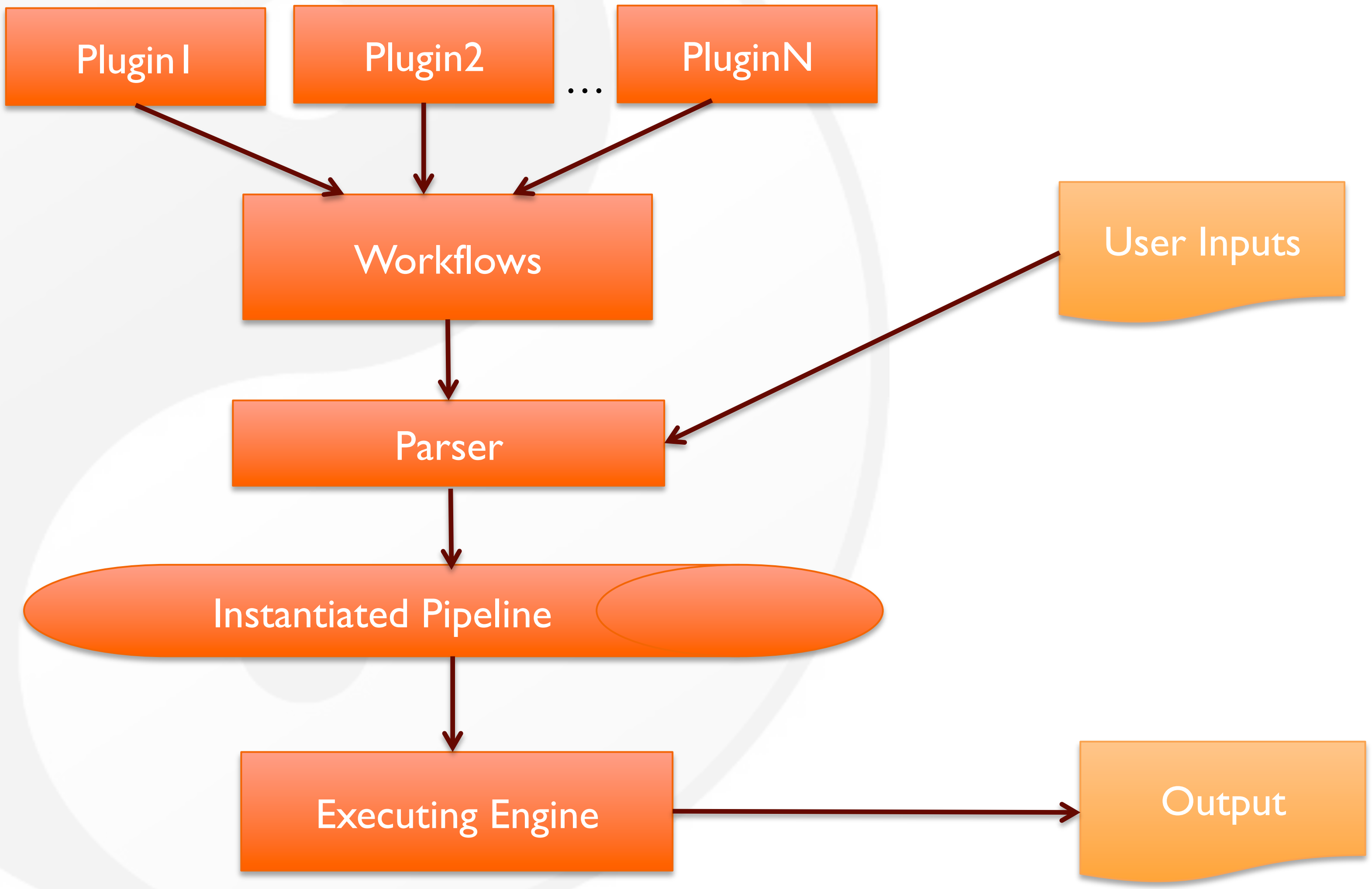
Roadmap

- Cancer Genome Research
  - Goals
  - Challenges of Next-Generation Sequencing
  - Towards Computing the Cure for Cancer (Phase I)
    - Open Genomics Engine (OpenGE)
- OpenGE
  - Overview
  - Workflow & Plug-In Specification
  - User Interface
  - Beyond OpenGE

OpenGE Design Goals

- Flexible
  - Support majority of existing genomics analysis tools
  - Allow composing sophisticated workflows
- Extensible
  - Fine-grained control of heterogeneous resources
    - Mapping between plugins and GPUs
    - Establish pipeline between CPU and GPUs
- Easy to Use
  - Lightweight
  - Currently provides intuitive command line interface
  - Could be extended to GUI in the future

OpenGE Overview



Plugin XML Definition

- Inspired by Galaxy
- Structures
  - Command(s)
  - Input parameters
  - Output parameters
- Conditional parameters
  - Ternary operator [condition? para1: para2]
    - String comparison
      - Str1 == Str2
      - Str1 != Str2
    - Boolean variables
      - True
      - False

```
<plugin id="bwa_aln" name="BWA Align" version="0.5.9">
   <description>Align reads with BWA</description>
   <commands>
      <command> bwa aln [$num_threads != ""? -t $numthreads]
$ref_genome $input_read -f $output_sai
      </command>
   </commands>

   <inputs>
      <param name="ref_genome" type="file" format="bwt_index"
label="Index of reference genome"/>
      <param name="input_read" type="file" format="fastq"
label="Input read file"/>
      <param name="num_threads" type="int" value="4"
label="Number of threads"/>
   </inputs>

   <outputs>
      <param name="output_sai" type="file" format="sai"
label="Output BWA alignments" />
   </outputs>
</plugin>
```

VirginiaTech
Invent the Future

Workflow XML Definition

- Essentially a directed acyclic graph (DAG) of plugins
- Structure
  - Inputs
  - Outputs
  - Steps
    - Plugin/sub-workflow
    - Inputs
    - Outputs
- Dependencies
  - Express dependency via input-output connections between steps
  - Output file automatically generated

Example Workflow

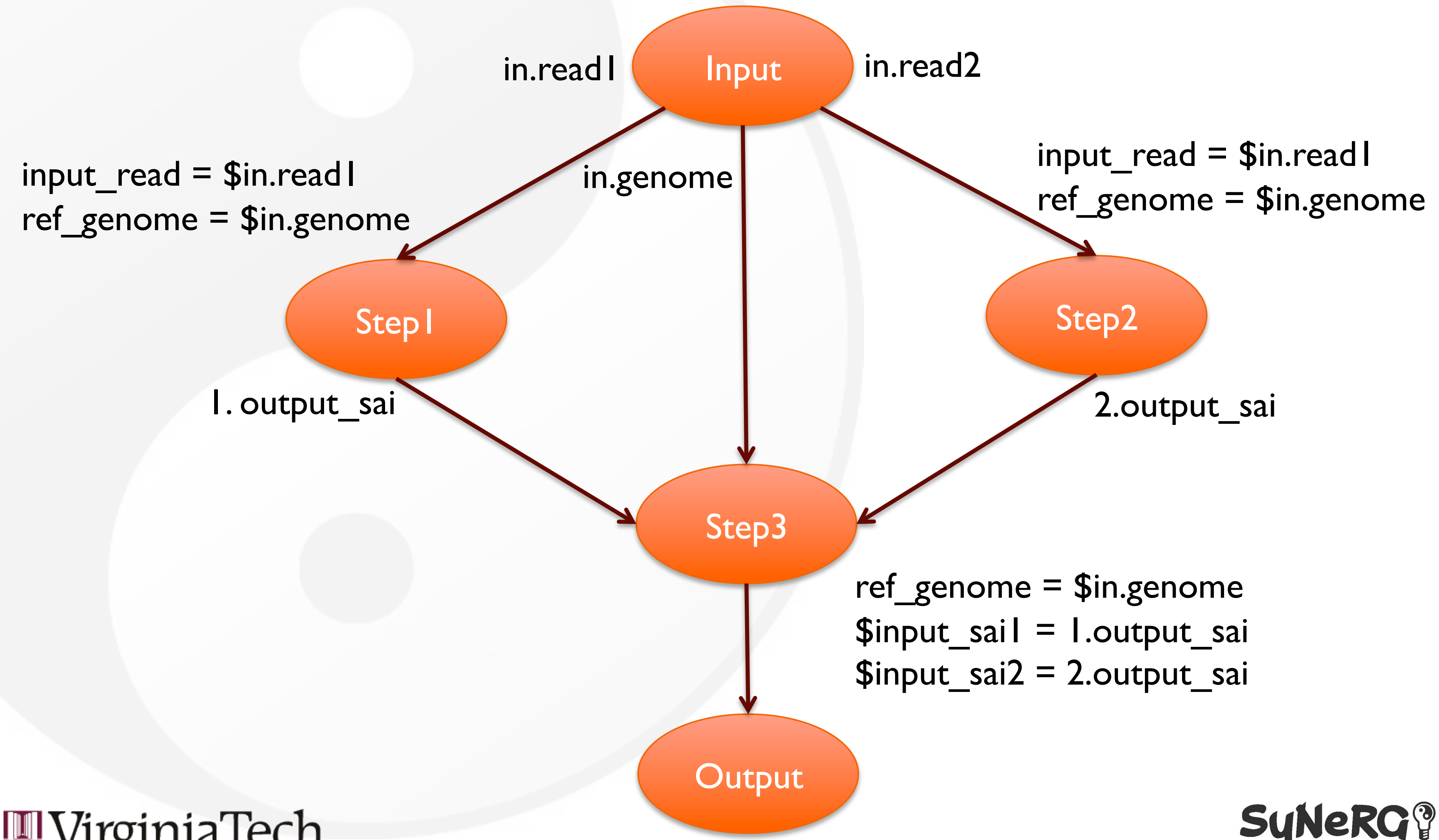
```
<inputs>
    <param name="in.read1" type="file" format="fastq" />
    <param name="in.read2" type="file" format="fastq" />
    <param name="in.genome" type="file" format="bwt" />
</inputs>

<steps>
    <step id="1" type="plugin" plugin_id="bwa_aln" >
        <inputs>
            <param name="input_read" value="$in.read1" />
            <param name="ref_genome" value="$in.genome" />
        </inputs>
        <outputs>
            <param name="output_sai" />
        </outputs>
    </step>
    <step id="2" type="plugin" plugin_id="bwa_aln" >
        <inputs>
            <param name="input_read" value="$in.read2" />
            <param name="ref_genome" value="$in.genome" />
        </inputs>
        <outputs>
            <param name="output_sai" />
        </outputs>
    </step>
    <step id="3" type="plugin" plugin_id="bwa_sampe" >
        <inputs>
            <param name="input_read1" value="$in.read1" />
            <param name="input_read2" value="$in.read2" />
            <param name="ref_genome" value="$in.genome" />
            <param name="input_sai1" value="$1.output_sai" />
            <param name="input_sai2" value="$2.output_sai" />
        </inputs>
        <outputs>
            <param name="output_sam" />
        </outputs>
    </step>
</steps>

<outputs>
    <param name="output_sam" type="file" format="sam"
value="$3.output_sam" />
</outputs>
```

synergy.cs.vt.edu

Workflow DAG



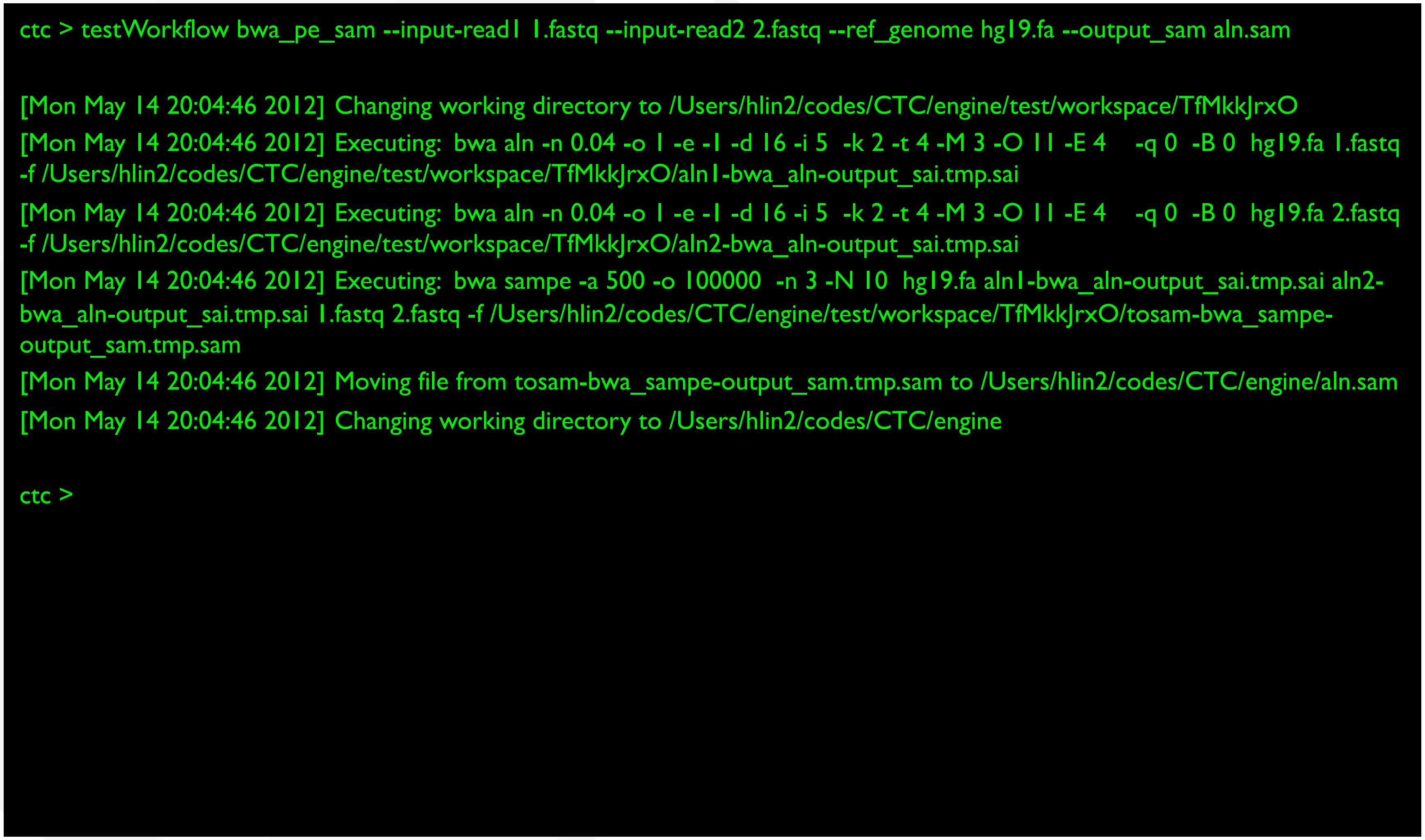
OpenGE User Interface

- Command line interface
- Programmable interface
- Annotated script importer

Command Line Interface

- Query
  - listWorkflows
  - listPlugins
  - queryWorkflow
  - queryPlugin
  - …
- Edit
  - CreatePluginTemplate
  - CreateWorkflow
  - …
- Execute
  - testWorkflow
  - executeWorkflow
  - …

CLI Screen Shot



```
ctc > testWorkflow bwa_pe_sam --input-read1 1.fastq --input-read2 2.fastq --ref_genome hg19.fa --output_sam aln.sam

[Mon May 14 20:04:46 2012] Changing working directory to /Users/hlin2/codes/CTC/engine/test/workspace/TfMkkJrxO
[Mon May 14 20:04:46 2012] Executing:  bwa aln -n 0.04 -o 1 -e -1 -d 16 -i 5  -k 2 -t 4 -M 3 -O 11 -E 4    -q 0  -B 0  hg19.fa 1.fastq
-f /Users/hlin2/codes/CTC/engine/test/workspace/TfMkkJrxO/aln1-bwa_aln-output_sai.tmp.sai
[Mon May 14 20:04:46 2012] Executing:  bwa aln -n 0.04 -o 1 -e -1 -d 16 -i 5  -k 2 -t 4 -M 3 -O 11 -E 4    -q 0  -B 0  hg19.fa 2.fastq
-f /Users/hlin2/codes/CTC/engine/test/workspace/TfMkkJrxO/aln2-bwa_aln-output_sai.tmp.sai
[Mon May 14 20:04:46 2012] Executing:  bwa sampe -a 500 -o 100000  -n 3 -N 10  hg19.fa aln1-bwa_aln-output_sai.tmp.sai aln2-
bwa_aln-output_sai.tmp.sai 1.fastq 2.fastq -f /Users/hlin2/codes/CTC/engine/test/workspace/TfMkkJrxO/tosam-bwa_sampe-
output_sam.tmp.sam
[Mon May 14 20:04:46 2012] Moving file from tosam-bwa_sampe-output_sam.tmp.sam to /Users/hlin2/codes/CTC/engine/aln.sam
[Mon May 14 20:04:46 2012] Changing working directory to /Users/hlin2/codes/CTC/engine

ctc >
```

VirginiaTech
Invent the Future

synergy.cs.vt.edu

Programmable Interface

```
Workflow workflow;

// Construct inputs of the workflow
Parameter p1(DATA_FILE, "", "fastq", "");
workflow.addInput("in_read1", p1);
....
// Construct steps of the workflow
WorkflowStep s_aln1(PLUGIN, "aln1", "bwa_aln");
s_aln1.addInput("input_read", "$in_read1");
s_aln1.addInput("ref_genome", "$in_genome");
s_aln1.addOutput("output_sai");
workflow.addStep(s_aln1);
...
 WorkflowStep s_aln2(PLUGIN, "aln2", "bwa_aln");
s_aln2.addInput("input_read", "$in_read2");
s_aln2.addInput("ref_genome", "$in_genome");
s_aln2.addOutput("output_sai");
workflow.addStep(s_aln2);
```

```
...
WorkflowStep s_tosam(PLUGIN, "tosam", "bwa_sampe");
s_tosam.addInput("input_read1", "$in_read1");
s_tosam.addInput("input_read2", "$in_read2");
s_tosam.addInput("ref_genome", "$in_genome");
s_tosam.addInput("input_sai1", "$aln1.output_sai");
s_tosam.addInput("input_sai2", "$aln2.output_sai");
 s_tosam.addOutput("output_sam");
 workflow.addStep(s_tosam);

Parameter p4(DATA_FILE, "$tobam.output_bam", "bam",
"");
workflow.addOutput("output", p4);
...
Engine engine(engine_dir);
engine.executeWorkflow(workflow, paras, true);
```

Annotated Scripts

- Import from users' existing workflow scripts
  - Automatically generate XML plugins and workflows
  - Automatically connect two consecutive steps
- Limitation
  - Support single input and single output for each step
- Inspired by Bpipe

  http://code.google.com/p/bpipe/

```
WORKFLOW_ID=imported_variant_calling
WORKFLOW_NAME="Call variants with samtools"
WORKFLOW_VERSION=1.0.0

REFERENCE=hg19.fa
align := {
   bwa aln -I -t 8 $REFERENCE $input > ${input}.sai
   bwa samse $REFERENCE ${input}.sai $input > $output
}
sort := {
   samtools view -bSu $input  | samtools sort - $output
   mv ${output}.bam ${output}
}
index := {
   samtools index $input
}
call_variants := {
   samtools mpileup -uf $REFERENCE $input | bcftools
view -bvcg - > $output
}
```

Acknowledgements

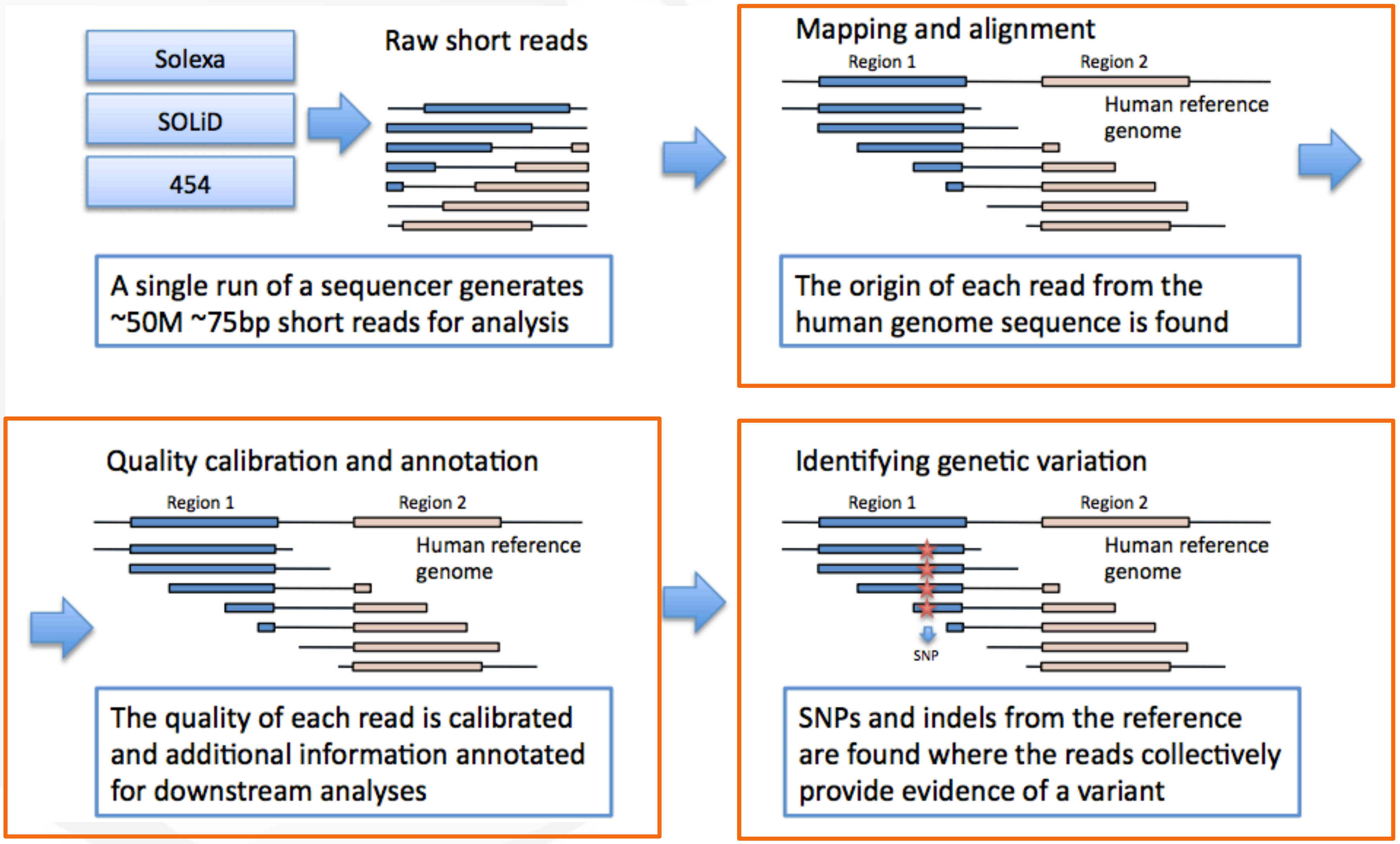
- David Mittelman, PhD, Assoc. Prof. @ VBI
  - Guidance on the life science aspects for the project
  - Caretaker of OpenGE
    - Future correspondence and questions on OpenGE to be forwarded to him
- Kenneth Lee and Jing Zhang
  - Contributions to FastR and the "Compute the Cure" framework → Open Genomics Engine (OpenGE)
- Gareth Highman
  - Contributions to RepeatSeq
- Ashwin Aji, NVIDIA Graduate Fellow
  - Contributions to GPU-accelerated dindel

Roadmap

- Cancer Genome Research
  - Goals
  - Challenges of Next-Generation Sequencing
  - Towards Computing the Cure for Cancer (Phase I)
    - Open Genomics Engine (OpenGE)
- OpenGE
  - Overview
  - Workflow & Plug-In Specification
  - User Interface
  - Beyond OpenGE: A Computer Scientist's Perspective

COMPUTE THE CURE

From Reads to Genetic Variation Detection



A single run of a sequencer generates ~50M ~75bp short reads for analysis

The origin of each read from the human genome sequence is found

The quality of each read is calibrated and additional information annotated for downstream analyses

SNPs and indels from the reference are found where the reads collectively provide evidence of a variant

VirginiaTech Invent the Future

Source: 1000 Genomes project: From mapping reads to de novo mutations, Mark DePristo, Broad Institute

SyNeRG synergy.cs.vt.edu

Read Mapping

- Problem definition
  - Given a read, identify where is from the reference genome

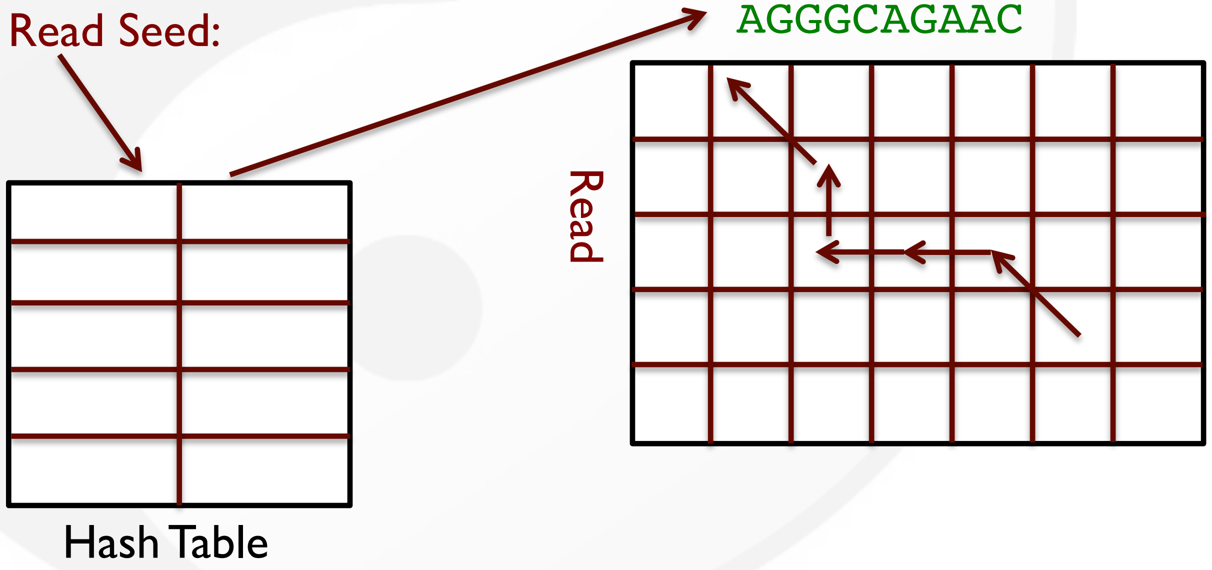
- Computational challenge?
  - Make it FAST … VERY FAST
    - Fastest short-read mapping algorithms take 13 CPU day to align a human genome with standard coverage
  - Make it accurate
    - Sequencing errors
    - Mapping errors

Hash-Based Mapping Algorithms

- Basic idea: Seed and extend
  - Build a hash table on k-length words on genome or reads
  - Segment query sequence into k-length seed words

Ref Genome: … CAAACCAGCTCTTAAGGGCAGAACTCTGAAAGACAACTGAGCTGCTG …

Read Seed: AGGGCAGAAC



Hash-Based Mapping Algorithms (Cont.)

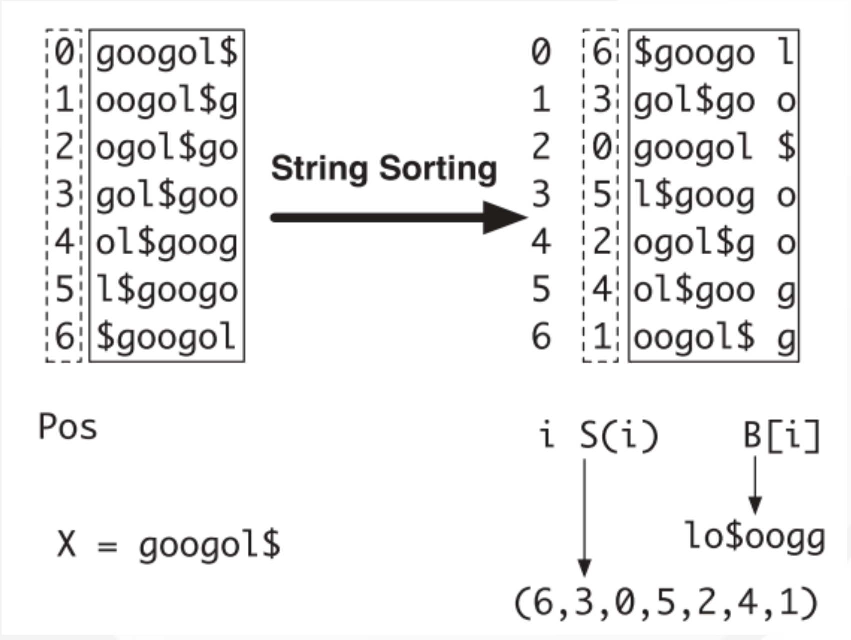
- Improvement: Spaced seeding
  - More sensitive than consecutive seeding

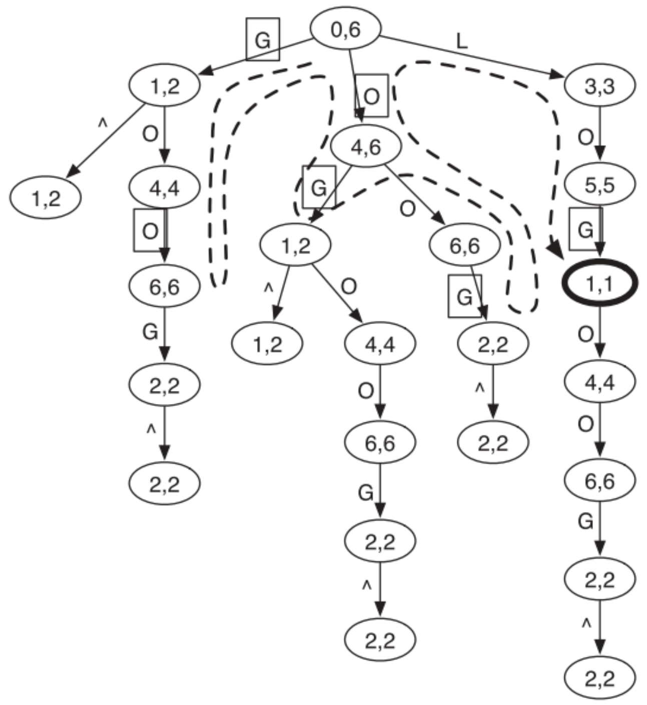
```
Ref Genome: … CAAACCAGCTCTTAAGGGCAGAACTCTGAAAGACAACTGAGCTGCTG …
Mask:                       100111110111
Read Seed:                  ATTGCAGACCTC
```

- Hashing strategies
  - Hash on reads
    - Memory efficient: controllable usage
    - Redundant computation for repetitive regions in the genome
  - Hash on genome
    - Save computation for searching repetitive regions
    - Memory intensive: 10s of GBs

FM-Index Based Mapping

- Build upon Burrows-Wheeler Transform
- Tree-based search → backward search ranges in suffix array
  - Mimic inexact search with exhaustive tree traversal





Source: Fast and Accurate Short Read Alignment with Burrows-Wheeler Transfer

synergy.cs.vt.edu

FM-Index Based Mapping (Cont.)

- Advantages
  - Small memory footprint
    - FM-Index: 2-8 GBs
    - Suffix tree: > 35 GBs
    - Suffix array: > 12 GBs
    - Hash-table: > 12 GBs
  - Fast mapping on repetitive regions

- Disadvantages
  - Search space grow fast as more mismatches and gaps allowed
  - Not applicable for long reads

FM-Index vs. Hash-Based Mapping

- FM-Index based mappers are widely used for speed
  - But less sensitive than hash-based approach
- Most accurate mappers are still hash-based
  - Examples: NovoAlign, Stampy

- Alignment tools used in the 1000 Genomes Project
  - Illumina: BWA (FM-Index)
  - ABI Solid: BFAST (Hash)
  - Roche 454: MOSAIK (Hash)

Emergent Trends

- Hybrid mapper
  - Use FM-Index based mappers to align well matched reads, and use hash-based mappers to align the rest
  - Example: Stampy

- FM-Index seed-and-extend mappers
  - Lookup seed matching in FM-Index
  - Extend seeded alignments with dynamic programming
  - Can be used to align long reads
    - Examples: BWA-SW, Bowtie2

Common Programming Components

- Indexing and lookup
  - Hashing with spaced seeding
  - FM-Index
- Dynamic programming
  - E.g., Smith-Waterman, Needleman-Wunsch
- Preliminary studies on GPU acceleration

	Applications	Speedup on GPU
Hashing on reads	RMAP	10 X
FM-Index	SOAP3	7.5 X over BWA
	CUSHAW	6-12 X over BWA
Smith Waterman	FastR (w/o traceback)	30 X
	FastR (w traceback)	7 X

Variation Discovery

- Opportunities
  - Abundance of parallelism (MapReduce type of computation)
    - Inference on each variant sites are independent
  - Early GPU acceleration study case
    - GSNP: 40X over SOAPsnp
- Challenges
  - Mapping statistical analysis on GPUs
  - Preliminary effort in accelerating DIndel with GPU
    - Detect short insertions and deletions in genome based probalistic realignments
    - Compute intensive: 18 hours on chromosome 22
    - Initial speedup: 2X
      - Bottleneck: data marshaling and demarshaling

Closing Thought

- A GPU-accelerated bioinformatics library for genome analysis?
  - Possible with convergence of algorithmic patterns

- Challenges
  - Bioinformatics algorithms are irregular
    - More challenging to map compared to dense matrix computation
    - Solution: Kepler?
  - What is the right level of abstractions
    - Balance between code restructuring and performance
    - Higher-level programming model to bridge the gap?

COMPUTE THE CURE

Conclusion

- Compute the Cure
  - A strategic philanthropic initiative of the NVIDIA Foundation that aims to support cancer researchers in the search for a cure.
- Open Genomics Engine (OpenGE)
  - An open-source software framework for cancer researchers to accelerate the identification of DNA mutations that lead to cancer.
- We Want You!
  - Open access to the OpenGE framework.
  - Source code repository to add algorithms and create plug-ins.
  - Seeking sponsors and adopters that may wish to connect OpenGE to their existing genomics workflow tools.