Applying In-Memory Technology to Genome Data Analysis

Cindy Fähnrich

Hasso Plattner Institute

GLOBAL HEALTH ’14 Tutorial

Hasso Plattner Institute
Key Facts

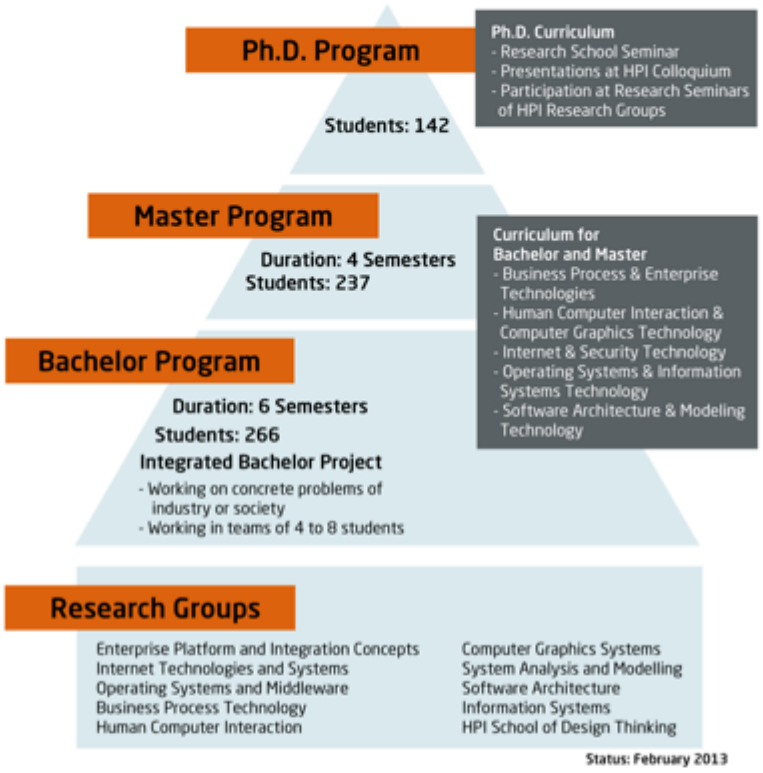
- Founded as a public-private partnership in 1998 in Potsdam near Berlin, Germany
- Institute belongs to the University of Potsdam
- Ranked 1st in CHE since 2009
- 500 B.Sc. and M.Sc. students
- 10 professors, 150 PhD students
- Course of study: IT Systems Engineering

In-Memory Applications For Informed Patients

Dr. Schapranow, HPI, Aug 12, 2014

Hasso Plattner Institute Programs

- Full university curriculum
- Bachelor (6 semesters)
- Master (4 semesters)
- Orthogonal Activities:
  - □ E-Health Consortium
  - □ School of Design Thinking
  - □ Research School



Ph.D. Program

Students: 142

Ph.D. Curriculum
- Research School Seminar
- Presentations at HPI Colloquium
- Participation at Research Seminars of HPI Research Groups

Master Program

Duration: 4 Semesters
Students: 237

Curriculum for Bachelor and Master
- Business Process & Enterprise Technologies
- Human Computer Interaction & Computer Graphics Technology
- Internet & Security Technology
- Operating Systems & Information Systems Technology
- Software Architecture & Modeling Technology

Bachelor Program

Duration: 6 Semesters
Students: 266
Integrated Bachelor Project
- Working on concrete problems of industry or society
- Working in teams of 4 to 8 students

Research Groups

Enterprise Platform and Integration Concepts
Internet Technologies and Systems
Operating Systems and Middleware
Business Process Technology
Human Computer Interaction
Computer Graphics Systems
System Analysis and Modelling
Software Architecture
Information Systems
HPI School of Design Thinking

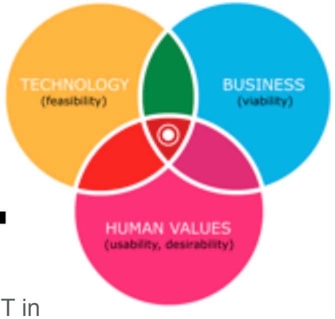
Status: February 2013

Hasso Plattner Institute
Enterprise Platform and Integration Concepts Group

Hasso Plattner Institut

Prof. Dr. h.c. Hasso Plattner

- Research focuses on the technical aspects of enterprise software and design of complex applications
  - □ In-Memory Data Management for Enterprise Applications
  - □ Enterprise Application Programming Model
  - □ Scientific Data Management
  - □ Human-Centered Software Design and Engineering
- Industry cooperations, e.g. SAP, Siemens, Audi, and EADS
- Research cooperations, e.g. Stanford, MIT, and Berkeley



Partner of Stanford Center for Design Research

Partner of MIT in Supply Chain Innovation and CSAIL

Partner at UC Berkeley RAD / AMP Lab

Partner of SAP AG

Agenda

1. **Introduction to In-Memory Technology**
2. Introduction to Genome Data Analysis
3. Combining In-Memory Technology with Genome Data Analysis
   - Pipeline Modeling
   - Pipeline Execution
   - IMDB Technology for Genome Data Analysis
   - IMDB Analysis Features for Applications

In-Memory Technology Building Blocks

- Combined column and row store
- Insert only for time travel
- Active/passive data store
- Dynamic multi-threading within nodes
- No aggregate tables
- On-the-fly extensibility
- Map reduce

- Minimal projections
- Bulk load
- Partitioning
- Analytics on historical data
- Single and multi-tenancy
- Object to relational mapping
- Group Key

- Any attribute as index
- Reduction of layers
- Multi-core/ parallelization
- SQL interface on columns & rows
- Lightweight Compression
- Text Retrieval and Extraction
- No disk

Combined Column and Row Store

- Row stores are designed for operative workload, e.g.
  - Create and maintain meta data for tests
  - Access a complete record of a trial or test series

- Column stores are designed for analytical work, e.g.
  - Evaluate the number of positive test results
  - Identification of correlations or test candidates

- In-Memory approach: Combination of both stores
  - Increased performance for analytical work
  - Operative performance remains interactively

Insert Only

- Traditional databases allow four data operations: INSERT, SELECT, DELETE, UPDATE

- DELETE and UPDATE are destructive since original data is no longer available

- Insert-only requires only first two to store a complete history (bookkeeping systems)

- Insert-only enables time travelling, e.g. to
  - Trace changes and reconstruct decisions
  - Document complete history of changes, therapies, etc.
  - Enable statistical observations

Lightweight Compression

- Main memory access is the new bottleneck

- Lightweight compression can reduce this bottleneck, i.e.
  - □ Improved usage of data bus capacity
  - □ Work directly on compressed data

recID	fname
…	…
39	John
40	Mary
41	Jane
42	John
…	…

Dictionary for "fname"

valueID	Value
…	…
23	John
24	Mary
25	Jane
…	…

Attribute Vector for "fname"

position	valueID
…	…
39	23
40	24
41	25
42	23
…	…

No Aggregate Tables

- IMDB paradigm: data stored at highest possible level of granularity

- Contrast to current practice of business data centers
  - Store on level of granularity required by application
  - Multiple applications use same data but require different granularity

→High data redundancy and maintenance efforts

- IMDB computes aggregates from source data on the

→Dramatical complexity decrease, easier maintenance

Partitioning

- Horizontal Partitioning
  - Cut long tables into shorter segments
  - E.g. to group samples with same relevance
- Vertical Partitioning
  - Split off columns to individual resources
  - E.g. to separate personalized data from experiment data
- Partitioning is the basis for
  - Parallel execution of database queries
  - Implementation of data aging and data retention management

Multi-Core and Parallelization

- Modern server systems consist of x CPUs, e.g.
  - Each CPU consists of y CPU cores, e.g. 8
  - Consider each of the x*y CPU core as individual workers
  - Each worker can perform one task at the same time in parallel

- Full table scan of database table w/ 1M entries results in 1/x*1/y search time when traversing in parallel
  - Reduced response time
  - No need for pre-aggregated totals and redundant data
  - Improved usage of hardware
  - Instant analysis of data

A ↔ P Active and Passive Data Store

- Active data are accessed frequently & updates are expected, e.g.
  - Most recent experiment results, e.g. last two weeks
  - Samples that have not been processed yet

- Passive data are used for analytical & statistical purposes, e.g.
  - Samples that were processed 5 years ago
  - Meta data about seeds that are not longer produced

- Moving passive data on slower storages
  - Reduces main memory demands
  - Improves performance for active data

Reduction of Application Layers

- Layers are introduced to abstract from complexity

- Each layer offers complete functionality, e.g. meta data of samples

- Less layer result in
  - Less code to maintain
  - More specific code
  - Reduced resource demands
  - Improves performance of applications due to eliminating obsolete processing steps

In-Memory Databases – History

- Original use case in 2006: Enterprise software
  - Combining operational and analytical data into one database
  - Enable real-time analysis on latest data

- Big data context: Business and accounting data, customer records, sales orders, invoices, …

- Started 2009 to use in-memory technology in the context of life sciences

- Big data context: Genomic/biological data, prescriptions, patient and cancer records, clinical information systems, medical publications, …

Agenda

1. Introduction to In-Memory Technology
2. **Introduction to Genome Data Analysis**
3. Combining In-Memory Technology with Genome Data Analysis
   - Pipeline Modeling
   - Pipeline Execution
   - IMDB Technology for Genome Data Analysis
   - IMDB Analysis Features for Applications

Precision Medicine – Motivation

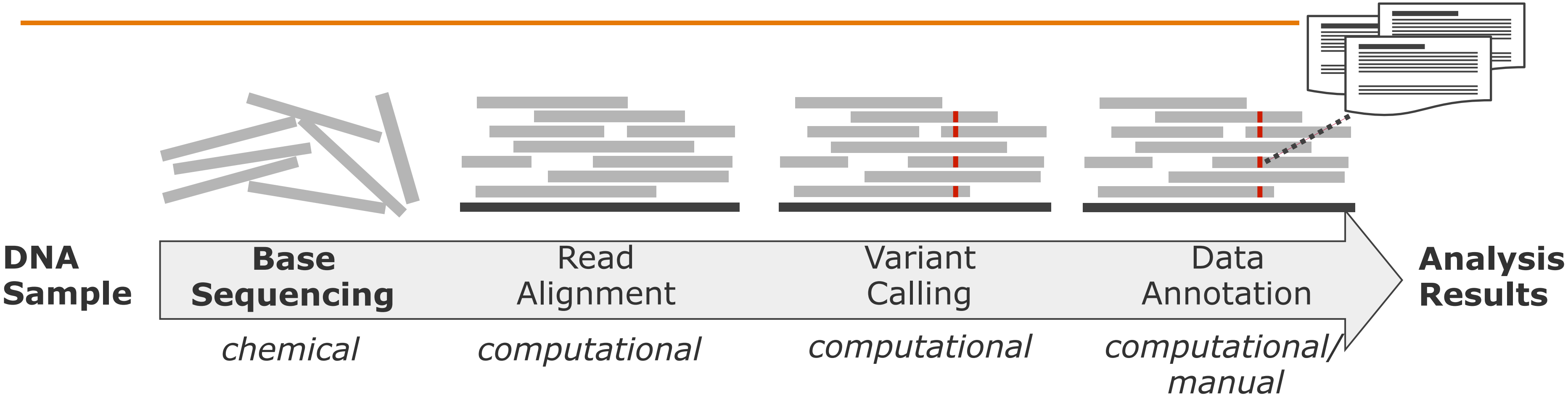
“Personalized medicine aims at treating patients specifically based on their individual dispositions, e.g. genetic or environmental factors”

(K. Jain, *Textbook of Personalized Medicine*. ***Springer, 2009)***

- Conventional cancer therapies often fail
  - □ One therapy does NOT fit all
  - □ Relation between genetic mutations and disease not considered/understood
  - → Analyze genetic profile of a patient to define customized therapies

- Challenge: Complex and time-consuming data processing tasks (analysis of one patient takes up to weeks)…

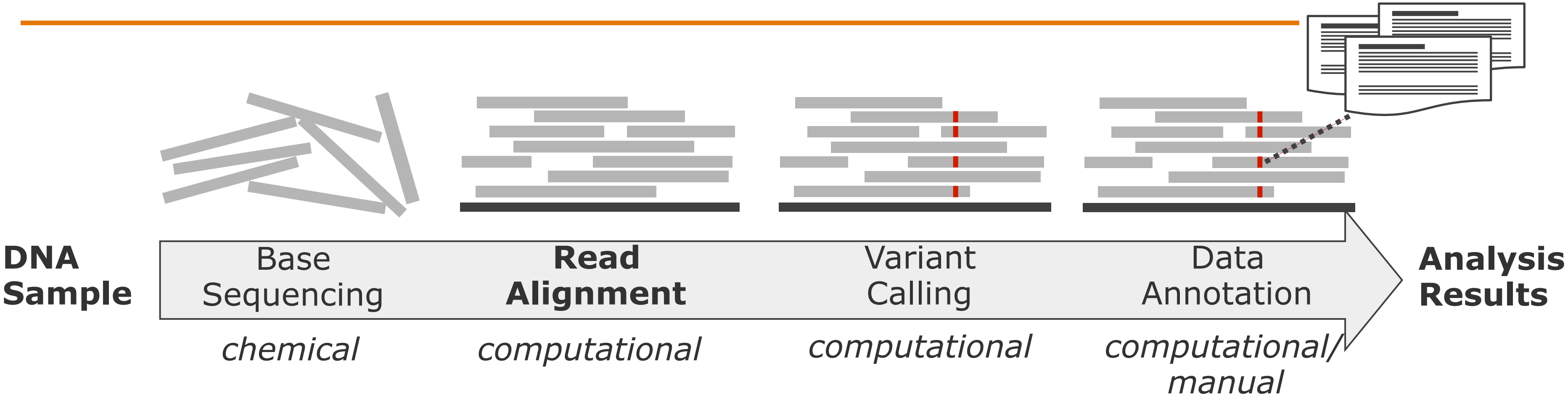
Genome Data Analysis



Base Sequencing

- Deriving DNA in digital format from sample via imaging procedures
- Output are unordered DNA snippets (=reads)
- High error rate → Sequencing at multiple coverage

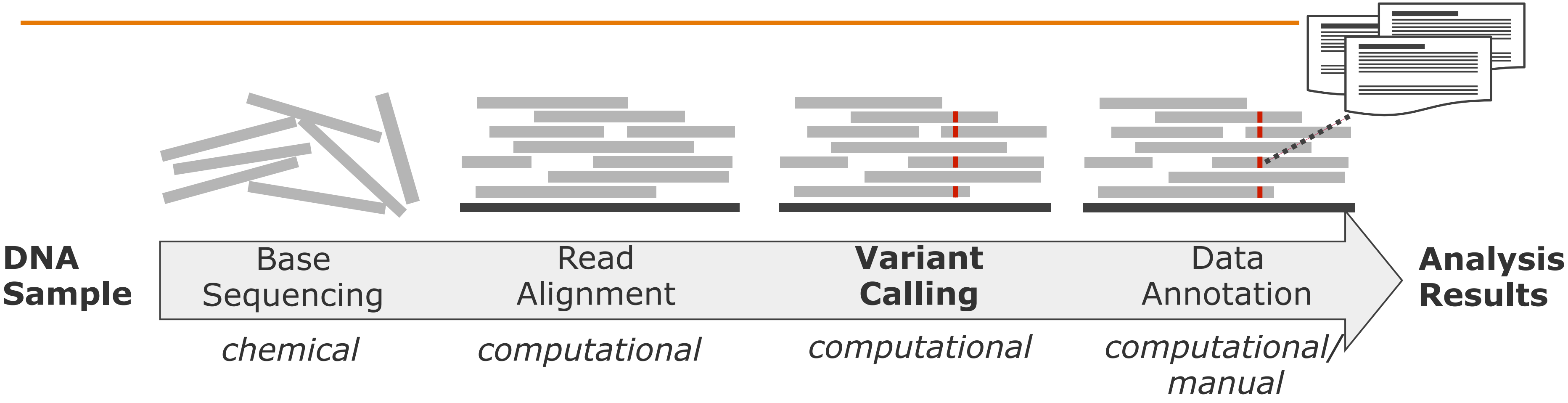
Genome Data Analysis



Alignment

- Reconstruct genome by reassembling all reads
- Pattern-matching vs. similarity search
- Matching strictness vs. runtime performance

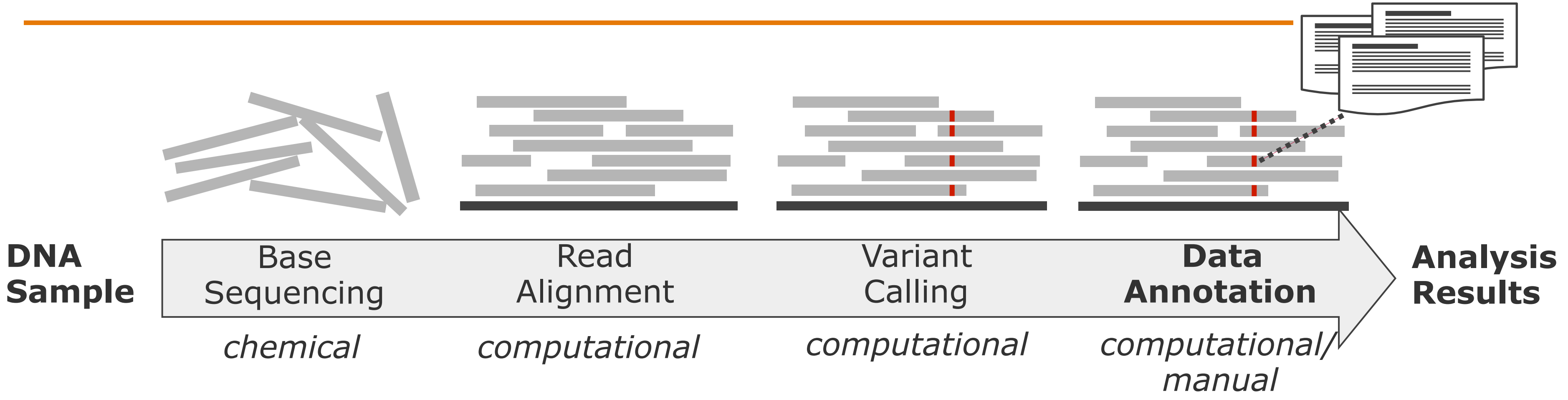
Genome Data Analysis



Variant Calling

- Detecting genetic variants in the sample genome
- Comparison to a reference
- Incorporating error probability of data

Genome Data Analysis



DNA Sample

Base Sequencing

chemical

Read Alignment

computational

Variant Calling

computational

Data Annotation

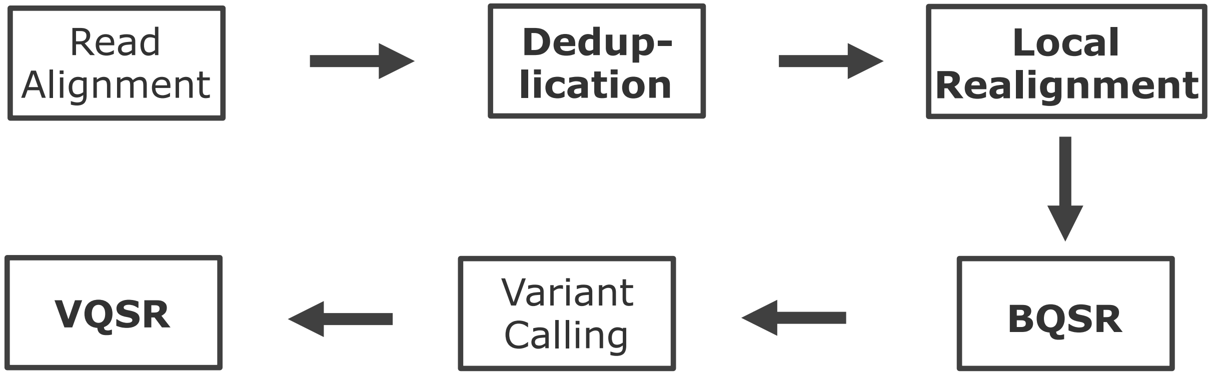
computational/ manual

Analysis Results

Data Annotation

- Find out impact of detected genetic variants on organism
- Connect known information, e.g. from studies/research papers, to genetic variants
- Gain new research insights, e.g. relations between genes and diseases, for personalized medicine

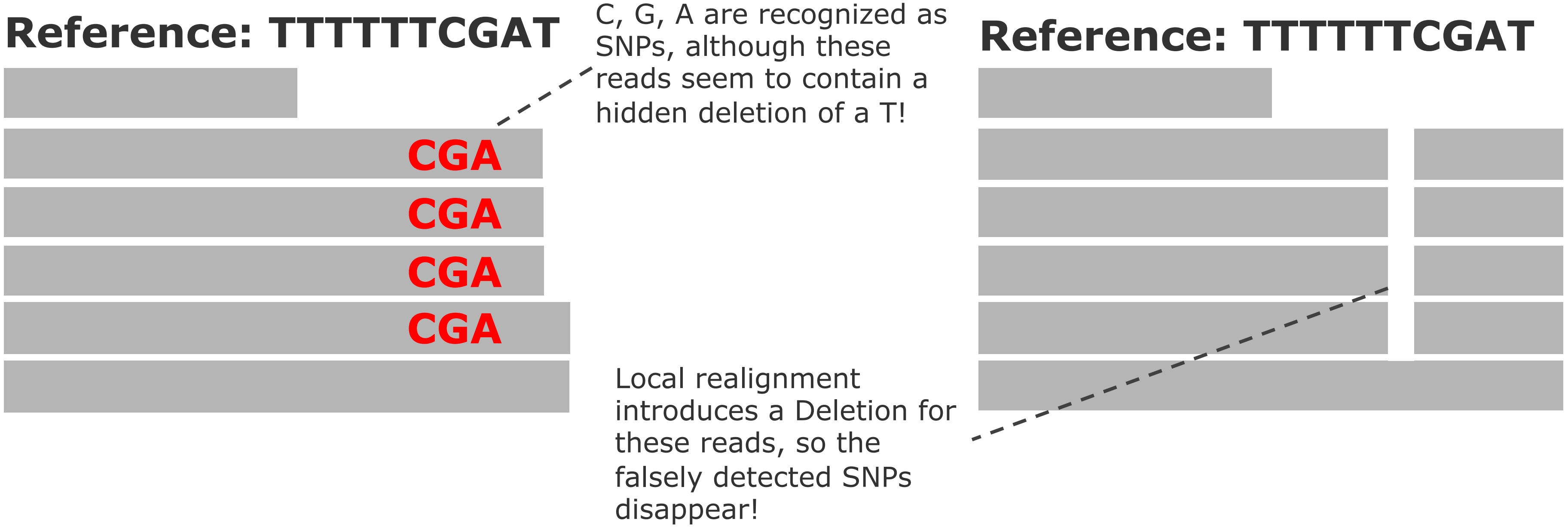
Genome Data Analysis – Alignment and Variant Calling Refined (1/3)



- Process requires intermediate steps to improve data quality
- Deduplication: Exclude duplicate reads from analysis
- Local Realignment: Reduce false posivites caused by Indels
- Base Quality Score Recalibration (BQSR): Adjust error probabilities of bases
- Variant Quality Score Recalibration (VQSR): Adjust variant probabilities

Additional Info: Local Realignment around InDels

- Insertions or Deletions (InDels) in reads can “trick” alignment algorithms into misaligning reads and introducing false positive Single Nucleotide Polimorphisms (SNPs)



Reference: TTTTTTCGAT

CGA

CGA

CGA

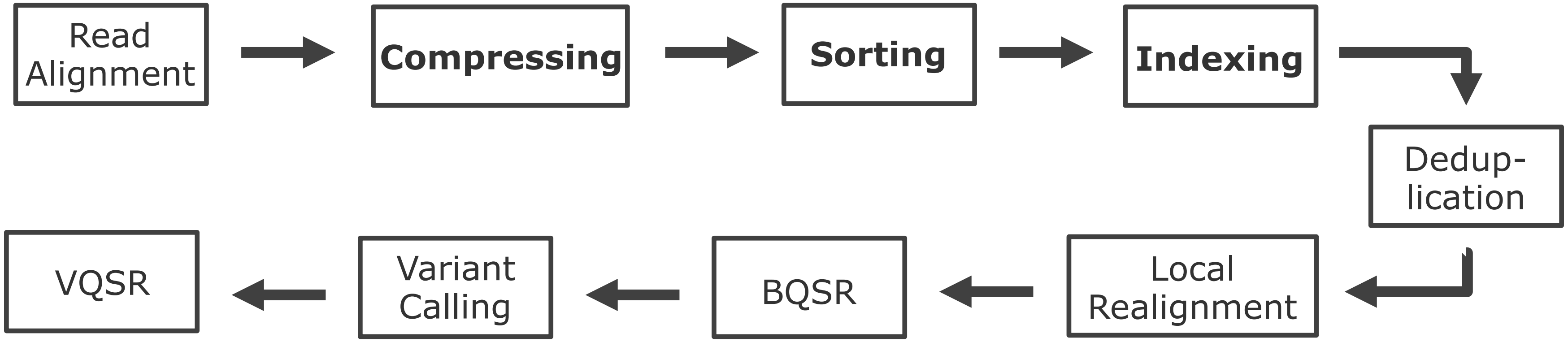
CGA

C, G, A are recognized as SNPs, although these reads seem to contain a hidden deletion of a T!

Reference: TTTTTTCGAT

Local realignment introduces a Deletion for these reads, so the falsely detected SNPs disappear!

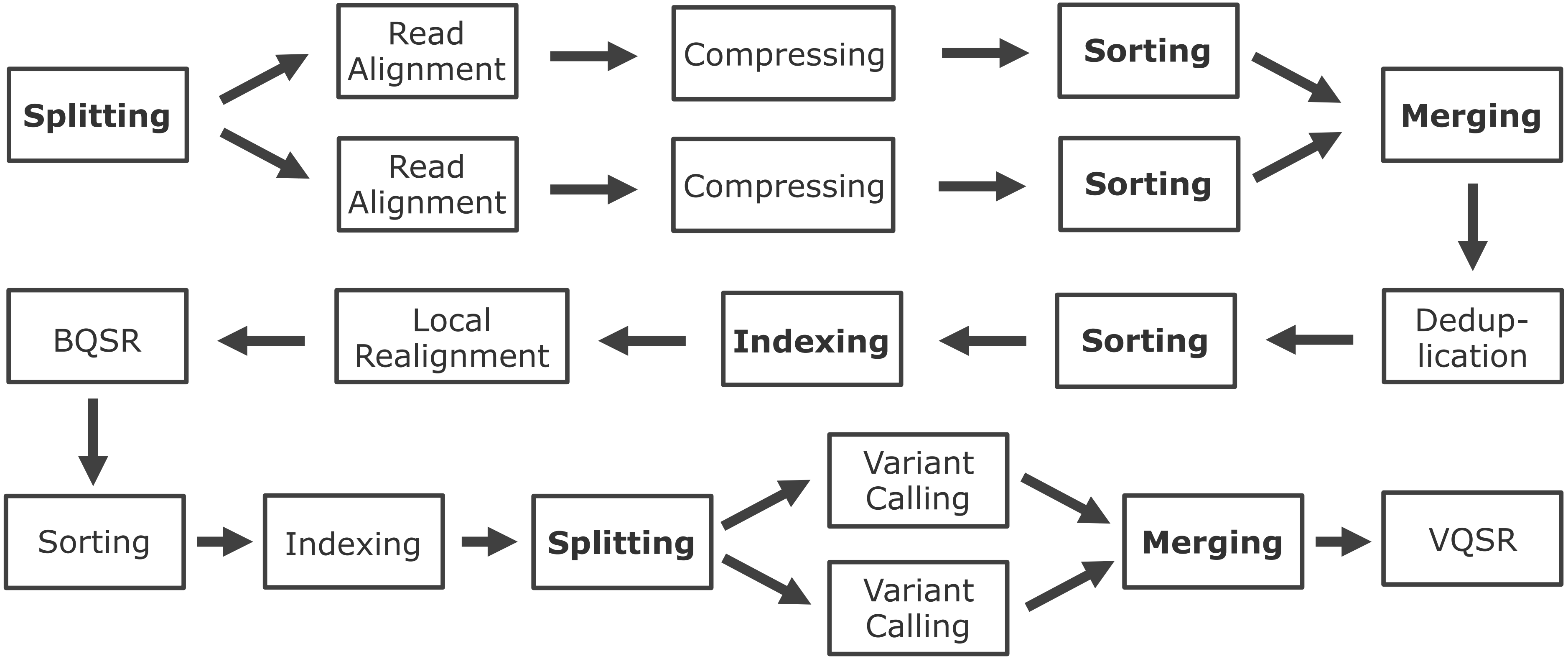
Genome Data Analysis – Alignment and Variant Calling Refined (2/3)



- Process requires intermediate steps to prepare data for faster processing
- More complicated when splitting up Alignment and Variant Calling …

Genome Data Analysis –
Alignment and Variant Calling Refined (3/3)

Hasso Plattner Institut



Genome Data Analysis – How it is done today

Alignment and Variant Calling

- Single tasks are triggered manually or in scripts invoking tools via command line:

```
bwa aln ref.fa sample.fastq| bwa samse ref.fa – sample.fastq | samtools view -Su - | samtools sort …
```

- Effective parallelization?
- Error handling?
- Distribution to a cluster?

Data Annotation

- Mostly manual analysis, e.g. via keyword search in portals on the web
- Efficient analysis of data from a patient/cohort?

Agenda

Hasso Plattner Institut

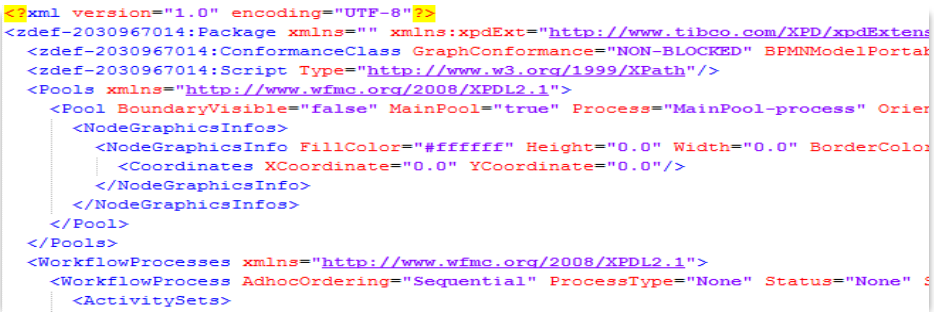
1. Introduction to In-Memory Technology
2. Introduction to Genome Data Analysis
3. **Combining In-Memory Technology with Genome Data Analysis**
   - **Pipeline Modeling**
   - Pipeline Execution
   - IMDB Technology for Genome Data Analysis
   - IMDB Analysis Features for Applications

Pipeline Modeling – How to Set Up a Pipeline

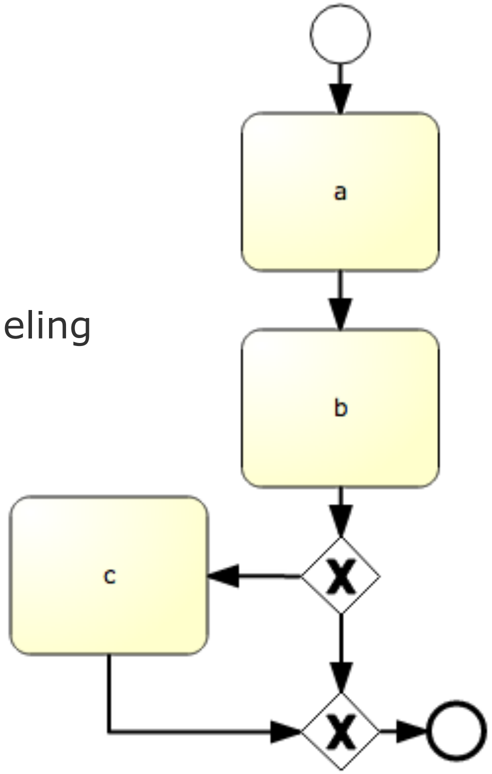
- Analysis pipeline is constructed from combining tools for the different analysis steps
  - Currently manual work via command line piping/scripts
  - Hard to understand/document/maintain
- Objective: Model the analysis pipeline with …
  - … a graphical representation that is …
  - … easy to understand and adapt
- Prerequisite: Graphical notation with standardized, machine readable representation

Pipeline Modeling – BPMN 2.0

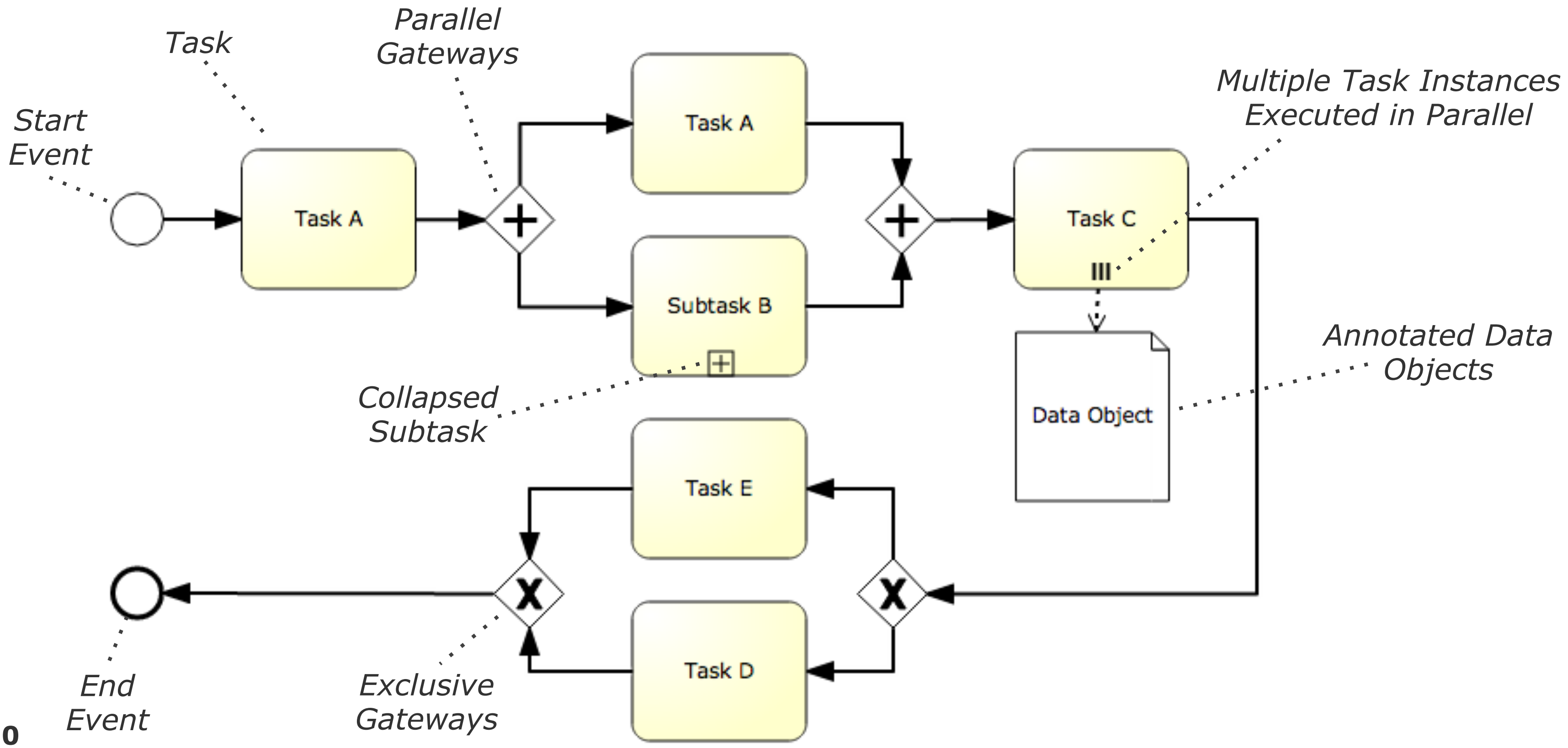
- Business Process Model and Notation (BPMN) 2.0
- Functional modeling of business processes and workflows
- Intended for both business and technical users → intuitive modeling
- XPDL available as XML standard for representing BPMN



```
<?xml version="1.0" encoding="UTF-8"?>
<zdef-2030967014:Package xmlns="" xmlns:xpdExt="http://www.tibco.com/XPD/xpdExtens
  <zdef-2030967014:ConformanceClass GraphConformance="NON-BLOCKED" BPMNModelPortab
  <zdef-2030967014:Script Type="http://www.w3.org/1999/XPath"/>
  <Pools xmlns="http://www.wfmc.org/2008/XPDL2.1">
    <Pool BoundaryVisible="false" MainPool="true" Process="MainPool-process" Orien
      <NodeGraphicsInfos>
        <NodeGraphicsInfo FillColor="#ffffff" Height="0.0" Width="0.0" BorderColor
          <Coordinates XCoordinate="0.0" YCoordinate="0.0"/>
        </NodeGraphicsInfo>
      </NodeGraphicsInfos>
    </Pool>
  </Pools>
  <WorkflowProcesses xmlns="http://www.wfmc.org/2008/XPDL2.1">
    <WorkflowProcess AdhocOrdering="Sequential" ProcessType="None" Status="None" S
      <ActivitySets>
```



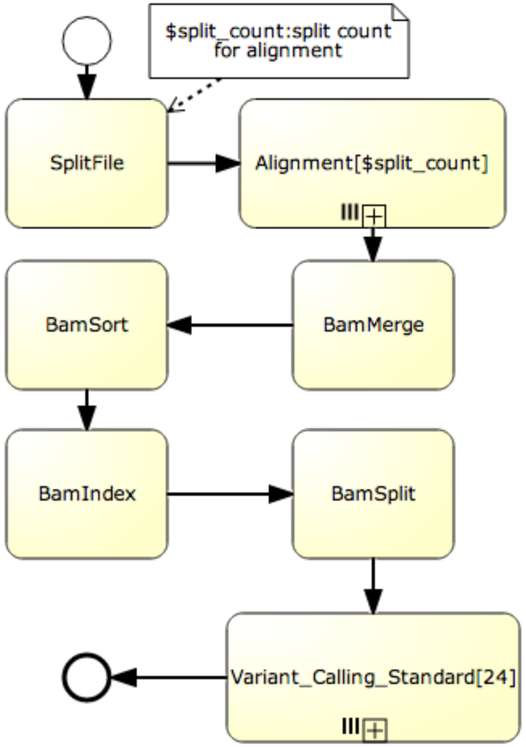
BPMN 2.0 – Basic Notation Overview



Pipeline Modeling with BPMN

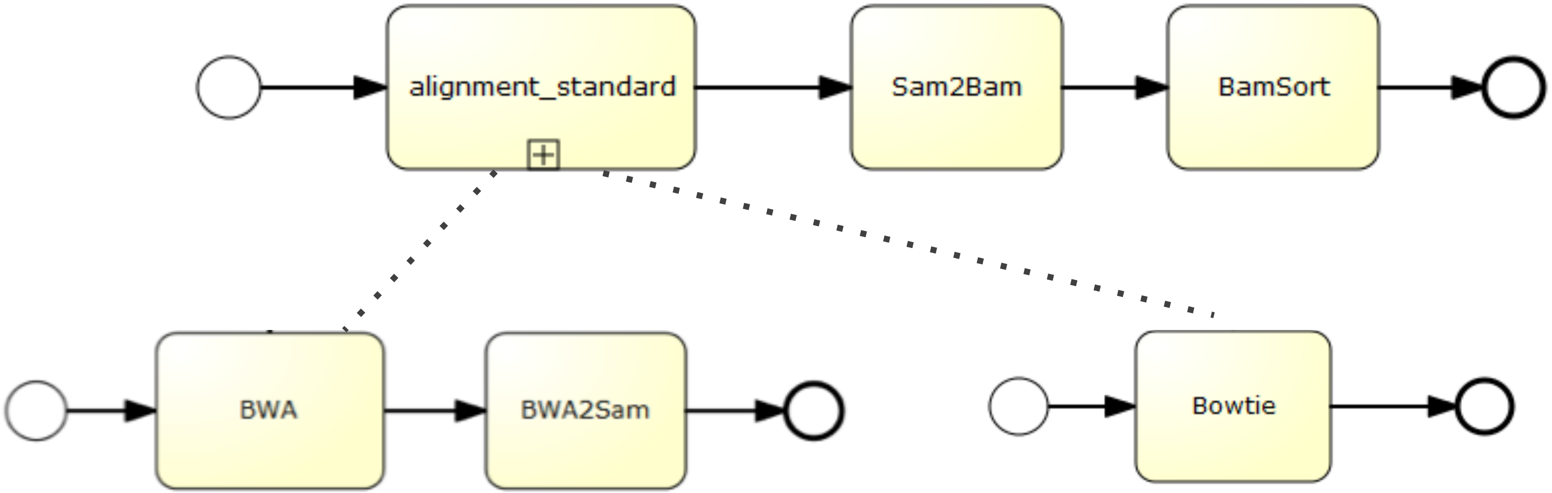
- Model and adapt your models in your tool of choice

- Only using a subset of BPMN, adapted with own constructs:
  - Modular structure
  - Degree of parallelization
  - Parameters
  - Variables


$split_count:split count for alignment
SplitFile
Alignment[$split_count]
BamSort
BamMerge
BamIndex
BamSplit
Variant_Calling_Standard[24]

Pipeline Modeling – Modular Structure

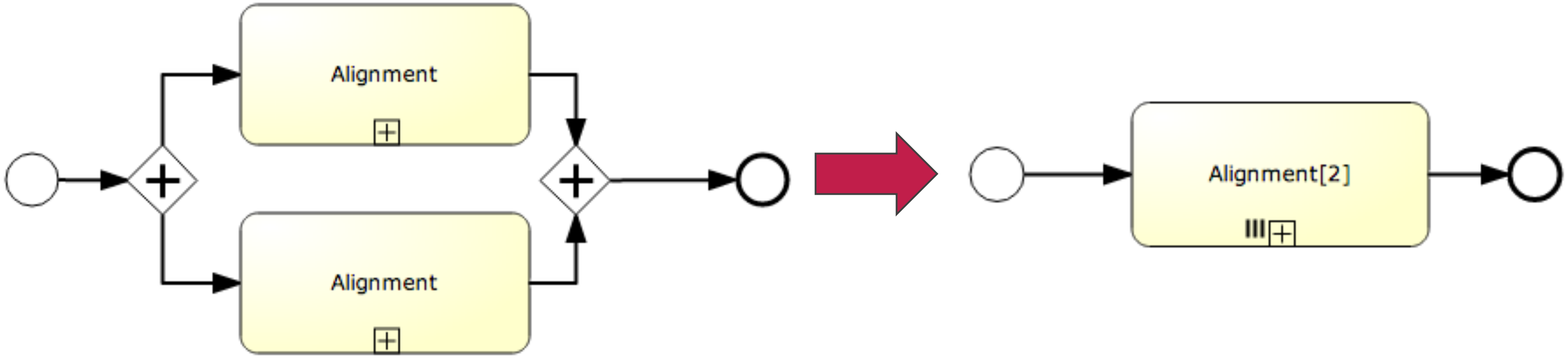
- Pipeline models can be nested hierarchically
- Reuse existing pipeline components, e.g. for alignment
- Make pipeline flexible regarding the tools used



Pipeline Modeling – Degree of Parallelization

- Execute parts of the pipeline in parallel

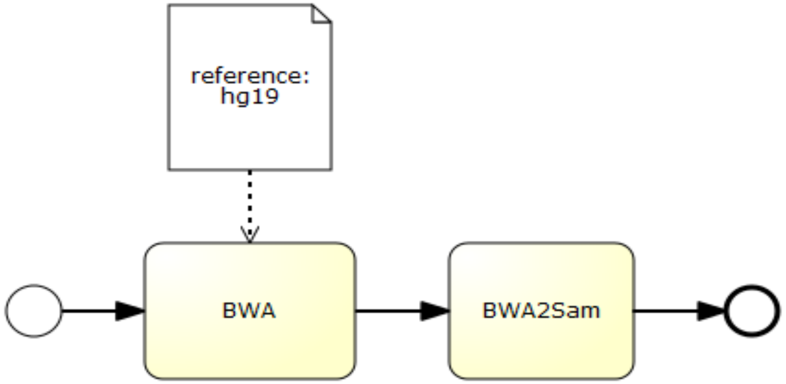
- Configure the explicit amount of parallel instances



Pipeline Modeling – Parameter

- Some tasks require parameters to be executed
  - Reference genome
  - Thread size
  - Number of parallel instances

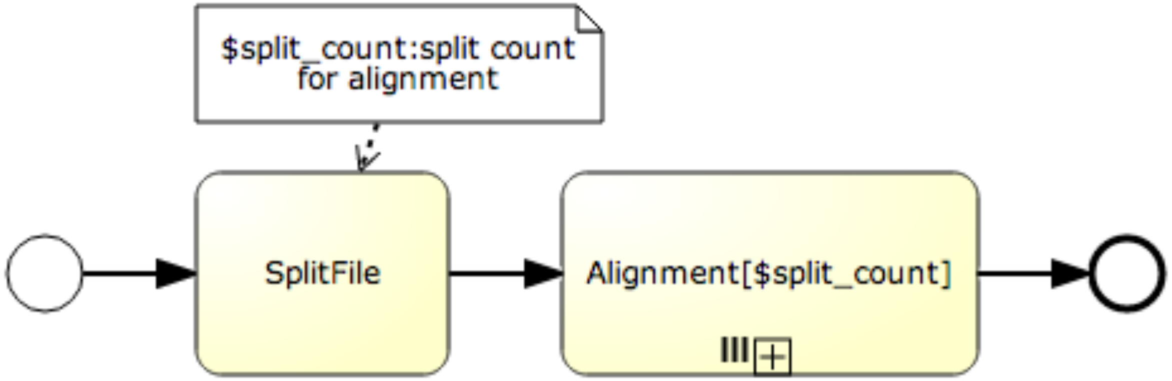
- Annotation of tasks with explicit parameters via data objects



Pipeline Modeling – Variables

- Some parameter values cannot be specified at design time
  - □ Number of parallel instances
  - □ Reference genome

- Annotate tasks with variables that are set at runtime



Pipeline Modeling – Creating the Final Analysis Pipeline

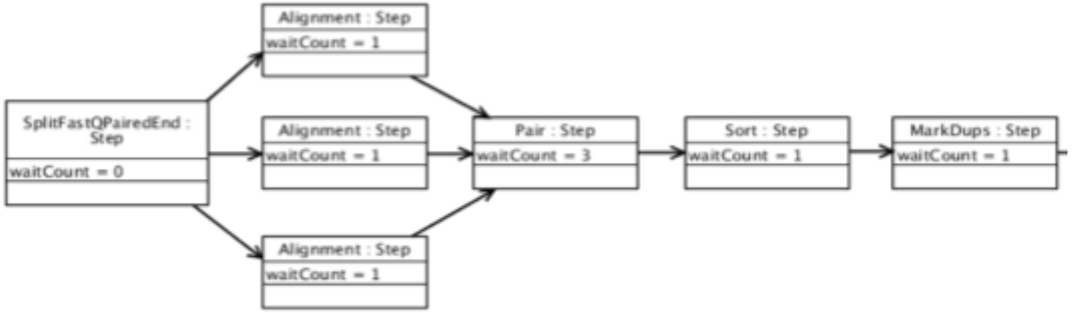
- Specify all subprocess models, parameters, variables
- Import all models in XPDL format into database
- Database entry of a pipeline model consists of
  - Name
  - Model ID
  - List of subprocess IDs
  - List of parameters and variables

Agenda

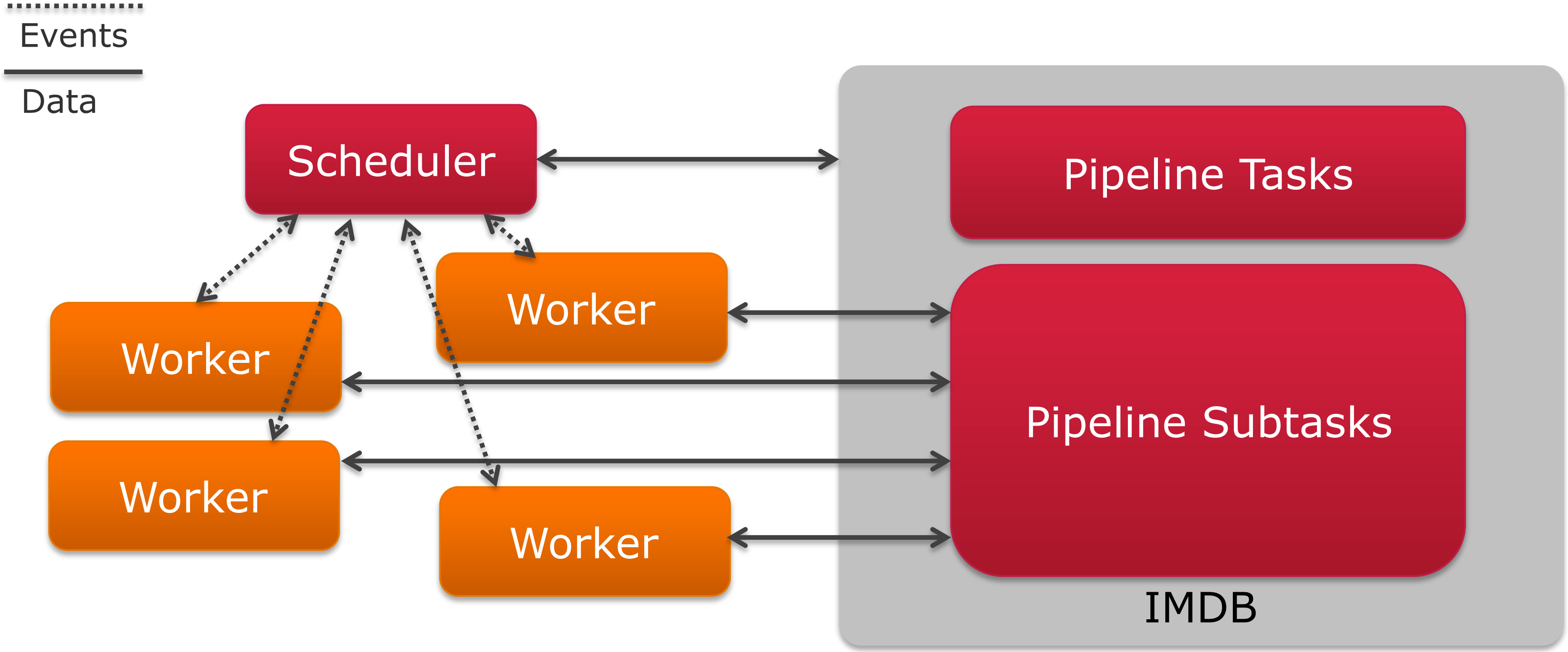
1. Introduction to In-Memory Technology
2. Introduction to Genome Data Analysis
3. **Combining In-Memory Technology with Genome Data Analysis**
   - Pipeline Modeling
   - **Pipeline Execution**
   - IMDB Technology for Genome Data Analysis
   - IMDB Analysis Features for Applications

Pipeline Execution –
Bringing the Model to Life

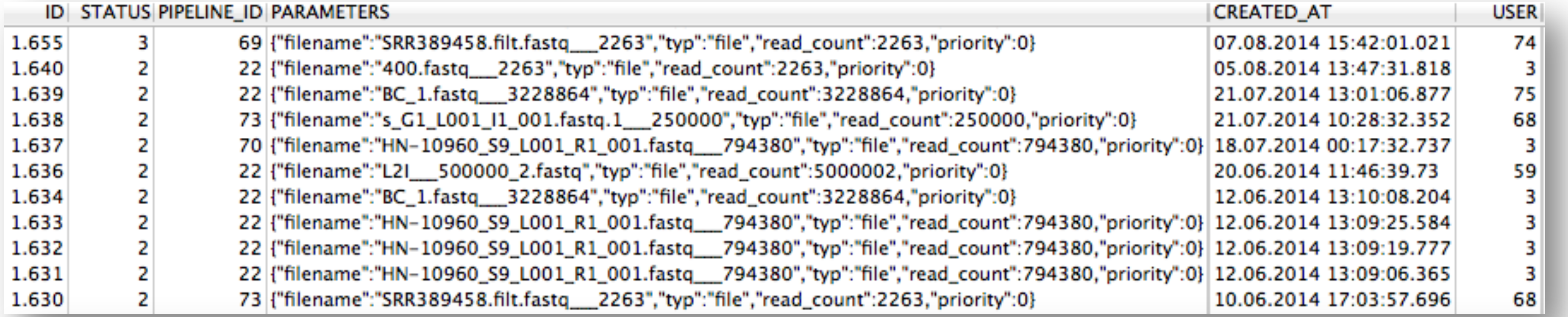
- Parser converts XML into executable, directed graph of task objects
- Execution environment is cluster of worker machines coordinated by scheduler
- Each task object has a corresponding implementation, e.g. tool invocation



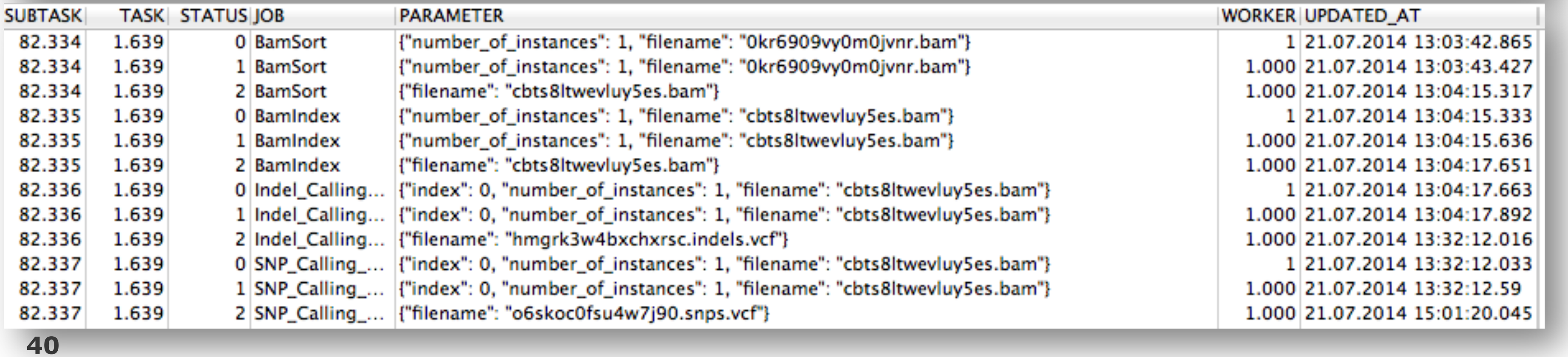
Pipeline Execution – Worker Framework



Pipeline Execution – Worker Framework



ID	STATUS	PIPELINE_ID	PARAMETERS	CREATED_AT	USER
1.655	3	69	{"filename":"SRR389458.filt.fastq___2263","typ":"file","read_count":2263,"priority":0}	07.08.2014 15:42:01.021	74
1.640	2	22	{"filename":"400.fastq___2263","typ":"file","read_count":2263,"priority":0}	05.08.2014 13:47:31.818	3
1.639	2	22	{"filename":"BC_1.fastq___3228864","typ":"file","read_count":3228864,"priority":0}	21.07.2014 13:01:06.877	75
1.638	2	73	{"filename":"s_G1_L001_I1_001.fastq.1___250000","typ":"file","read_count":250000,"priority":0}	21.07.2014 10:28:32.352	68
1.637	2	70	{"filename":"HN-10960_S9_L001_R1_001.fastq___794380","typ":"file","read_count":794380,"priority":0}	18.07.2014 00:17:32.737	3
1.636	2	22	{"filename":"L2I___500000_2.fastq","typ":"file","read_count":5000002,"priority":0}	20.06.2014 11:46:39.73	59
1.634	2	22	{"filename":"BC_1.fastq___3228864","typ":"file","read_count":3228864,"priority":0}	12.06.2014 13:10:08.204	3
1.633	2	22	{"filename":"HN-10960_S9_L001_R1_001.fastq___794380","typ":"file","read_count":794380,"priority":0}	12.06.2014 13:09:25.584	3
1.632	2	22	{"filename":"HN-10960_S9_L001_R1_001.fastq___794380","typ":"file","read_count":794380,"priority":0}	12.06.2014 13:09:19.777	3
1.631	2	22	{"filename":"HN-10960_S9_L001_R1_001.fastq___794380","typ":"file","read_count":794380,"priority":0}	12.06.2014 13:09:06.365	3
1.630	2	73	{"filename":"SRR389458.filt.fastq___2263","typ":"file","read_count":2263,"priority":0}	10.06.2014 17:03:57.696	68

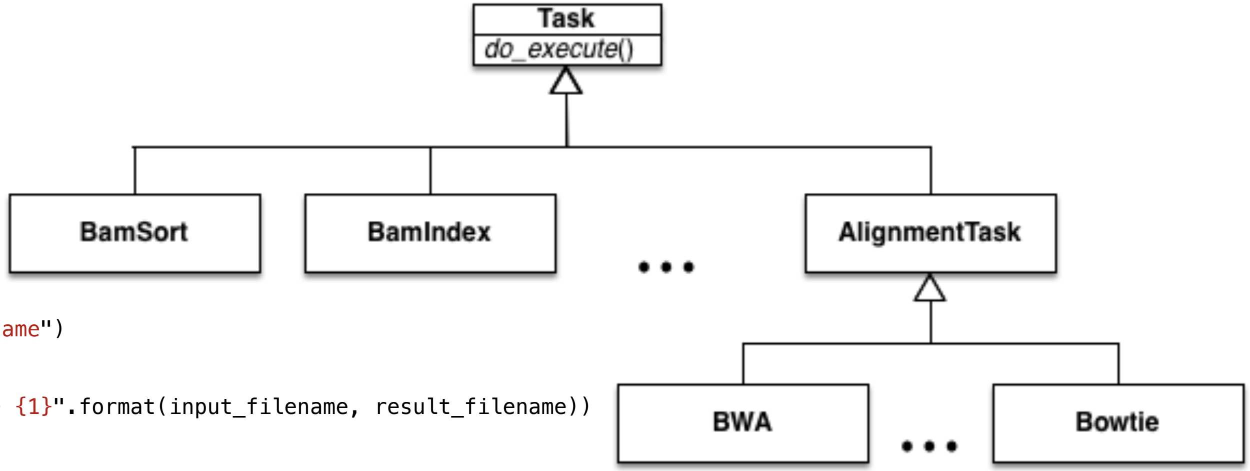


SUBTASK	TASK	STATUS	JOB	PARAMETER	WORKER	UPDATED_AT
82.334	1.639	0	BamSort	{"number_of_instances": 1, "filename": "0kr6909vy0m0jvnr.bam"}	1	21.07.2014 13:03:42.865
82.334	1.639	1	BamSort	{"number_of_instances": 1, "filename": "0kr6909vy0m0jvnr.bam"}	1.000	21.07.2014 13:03:43.427
82.334	1.639	2	BamSort	{"filename": "cbts8ltwevluy5es.bam"}	1.000	21.07.2014 13:04:15.317
82.335	1.639	0	BamIndex	{"number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1	21.07.2014 13:04:15.333
82.335	1.639	1	BamIndex	{"number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1.000	21.07.2014 13:04:15.636
82.335	1.639	2	BamIndex	{"filename": "cbts8ltwevluy5es.bam"}	1.000	21.07.2014 13:04:17.651
82.336	1.639	0	Indel_Calling...	{"index": 0, "number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1	21.07.2014 13:04:17.663
82.336	1.639	1	Indel_Calling...	{"index": 0, "number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1.000	21.07.2014 13:04:17.892
82.336	1.639	2	Indel_Calling...	{"filename": "hmgrk3w4bxchxrsc.indels.vcf"}	1.000	21.07.2014 13:32:12.016
82.337	1.639	0	SNP_Calling_...	{"index": 0, "number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1	21.07.2014 13:32:12.033
82.337	1.639	1	SNP_Calling_...	{"index": 0, "number_of_instances": 1, "filename": "cbts8ltwevluy5es.bam"}	1.000	21.07.2014 13:32:12.59
82.337	1.639	2	SNP_Calling_...	{"filename": "o6skoc0fsu4w7j90.snps.vcf"}	1.000	21.07.2014 15:01:20.045

Pipeline Execution – Task Implementation

- Task implementation imported as modules to worker at runtime

- One super class for administrative things, all tasks implement particular method



```
class BamSort(Task):

    def do_execute(self):
        input_filename = self.get_input("filename")

        result_filename = self.new_filename()
        self.system_command(”samtools sort {0} {1}".format(input_filename, result_filename))


        self.add_output("filename", result_filename)
```

Pipeline Execution – Scheduler

- Scheduler is responsible for holding the structure of task objects

Insert Only
For Time Travel

- Starts task when all predecessors are finished

Analytics on
Historical Data

t

- High availability of scheduler by storing global pipeline status in IMDB

→ In case of scheduler crash another worker can take scheduler role without any delay

- Scheduler uses workload information and execution statistics based on logs in IMDB

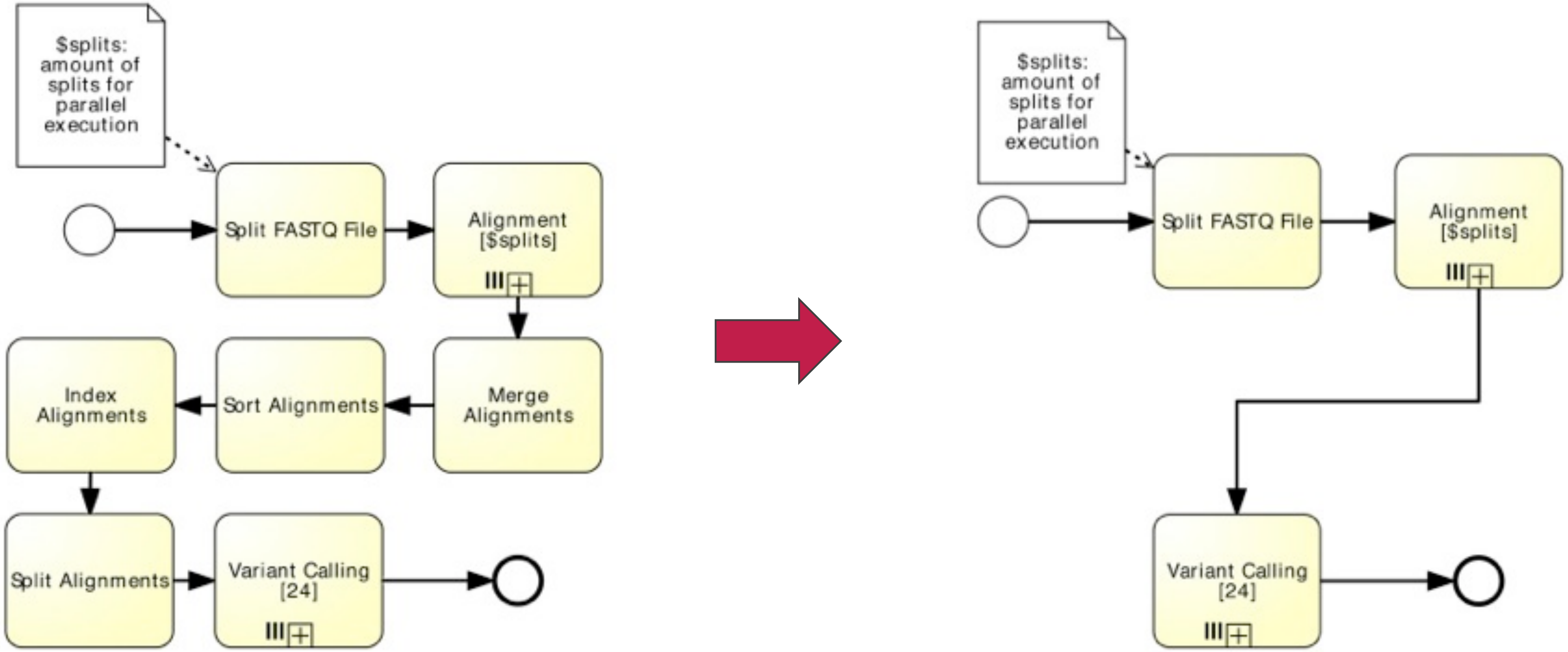
Pipeline Execution – Scheduling Policies

- Different scheduling algorithms
  - First-come first-served
  - Lottery
  - Shortest task first
  - Priority-based assignment
  - User-based assignment

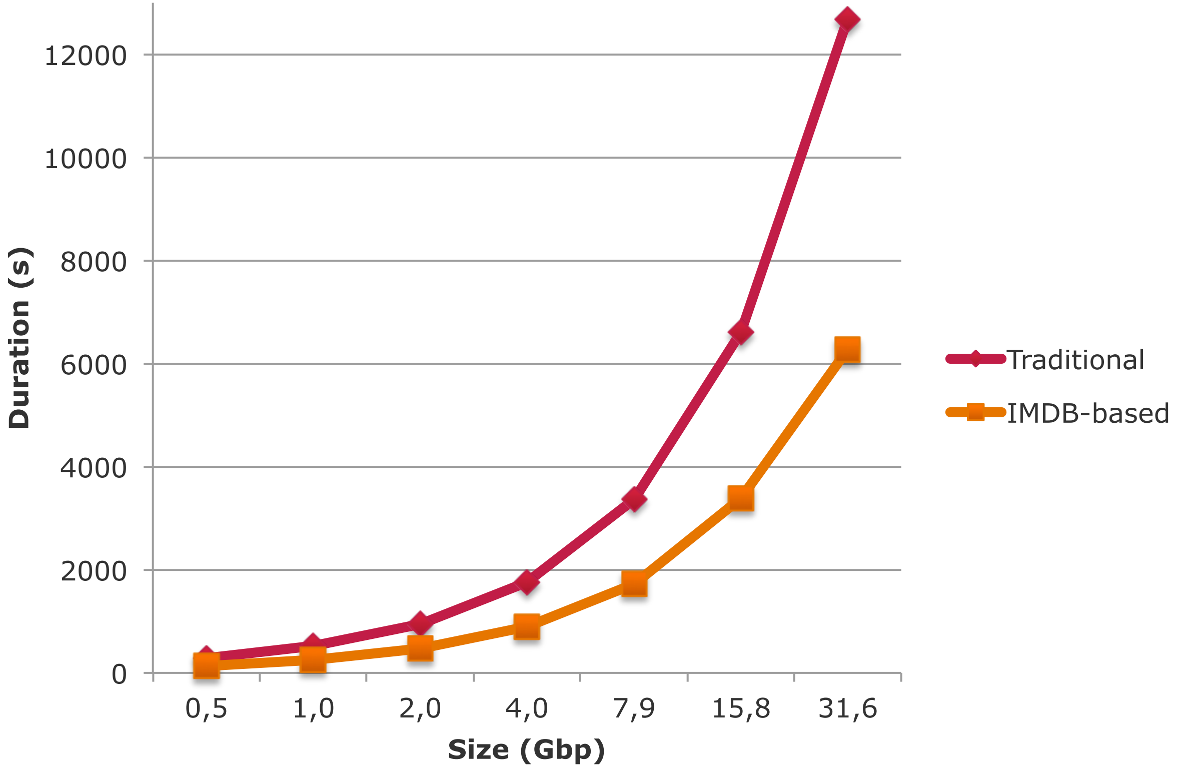
- Prioritize tasks to maximize utilization of workers

Pipelines –
Traditional vs. IMDB-supported Approach

- (Intermediate) results are imported into database



Pipelines – Traditional vs. IMDB-supported Approach

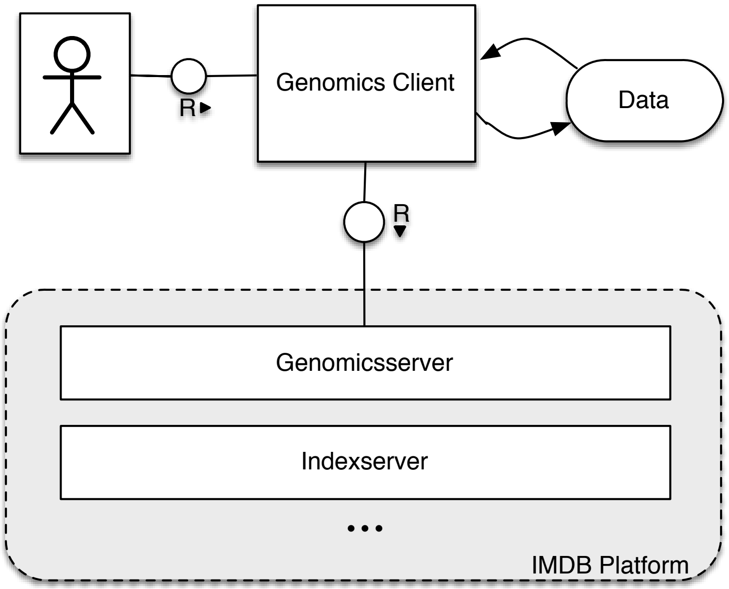


Agenda

1. Introduction to In-Memory Technology
2. Introduction to Genome Data Analysis
3. **Combining In-Memory Technology with Genome Data Analysis**
   - Pipeline Modeling
   - Pipeline Execution
   - **IMDB Technology for Genome Data Analysis**
   - IMDB Analysis Features for Applications

IMDB Technology for Genome Data Analysis – Alignment

Hasso Plattner Institut



- Extension of IMDB platform via own Genomicsserver
- Index creation at server start and storage in main memory
- Efficient processing via vectorization and bit parallelism
- Efficient streaming capabilities provided by IMDB platform

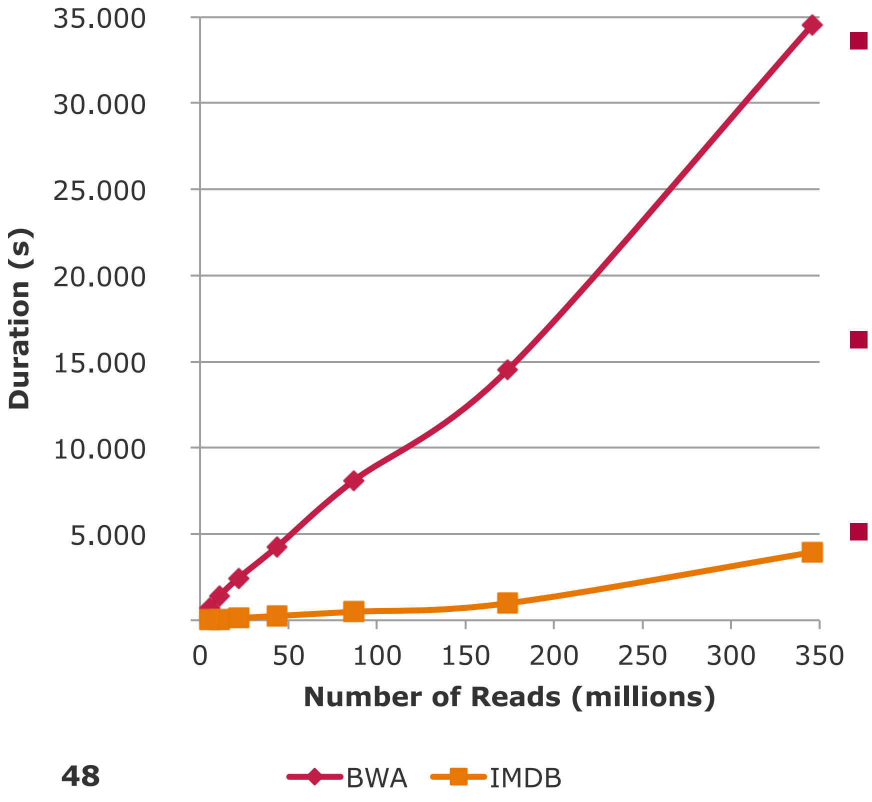
Multi-Core and Parallelization

Partitioning

Lightweight Compression

No Disk

Alignment – Performance



- First evaluations with Burrows-Wheeler-Alignment (BWA) as one representative of popular alignment algorithms

- Time saving up to a factor of 21 compared to BWA

- Alignment of low-coverage (20x) whole genome on cluster
  - □ Up to 346M reads
  - □ Alignment of all reads within ~1h

Variant Calling – Motivation

Hasso Plattner Institut

- Common variant calling tools all process files residing on disk space
  - Slow storage media
  - Large data files, e.g. >100GB per individual

- Idea: Access data from main memory and profit from built-in database features
  - Partitioning
  - Multi-core and parallelization
  - Lightweight compression

Multi-Core and Parallelization

Partitioning

Lightweight Compression

No Disk

Variant Calling – Data

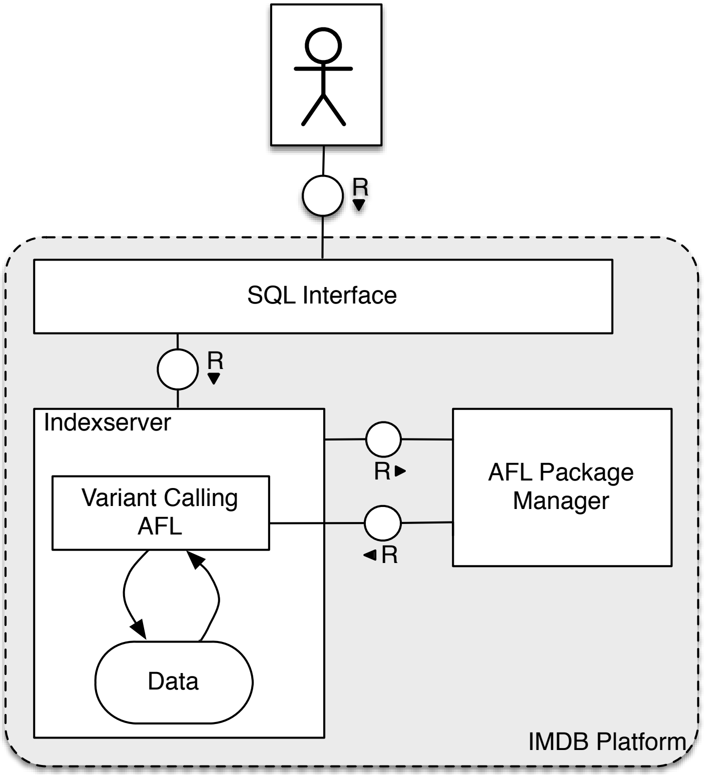
- Reference genome: Base sequence for comparison

- Read alignments: Reads from reconstructed sample genome

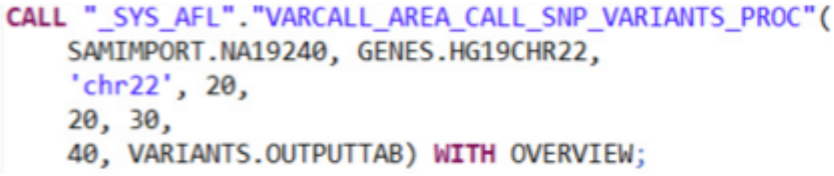
- All data is imported into database beforehand, with implicit
  - Data indexing
  - Lightweight compression

- Variant calling results conform to standard format and can easily be exported from database or used for further analyses

Variant Calling – Extending the Database Core

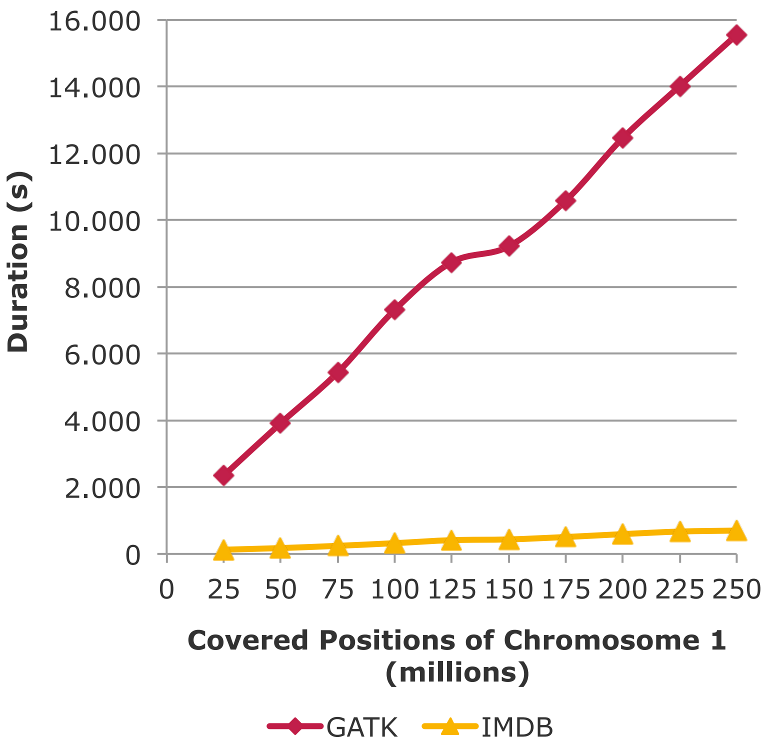


- Implementation as Application Function Library (AFL)
- Variant calling per chromosome
- Parallelization with MapReduce-like approach
- Invocation via stored procedure call



```
CALL "_SYS_AFL"."VARCALL_AREA_CALL_SNP_VARIANTS_PROC"(
    SAMIMPORT.NA19240, GENES.HG19CHR22,
    'chr22', 20,
    20, 30,
    40, VARIANTS.OUTPUTTAB) WITH OVERVIEW;
```

Variant Calling – Performance



- Built-in database functionalities simplify and speed up data preprocessing

- Average time saving of factor 22 compared to standard tools at equal accuracy

- SNP calling of high-coverage (64x) whole genome on cluster
  - □ 873M read alignments
  - □ ~18min

Agenda

1. Introduction to In-Memory Technology
2. Introduction to Genome Data Analysis
3. **Combining In-Memory Technology with Genome Data Analysis**
   - Pipeline Modeling
   - Pipeline Execution
   - IMDB Technology for Genome Data Analysis
   - **IMDB Analysis Features for Applications**

IMDB Analysis Features for Applications – Textual Analysis of Medical Documents

- IMDB provides text analysis features, e.g.
  - □ Fulltext indexing
  - □ Entity Recognition
  - □ Tokenization
  - □ Fuzzy search

- Mechanisms can be made domain-specific by specifying
  - □ Dictionaries
  - □ CGUL rules containing regular expressions with linguistic attributes

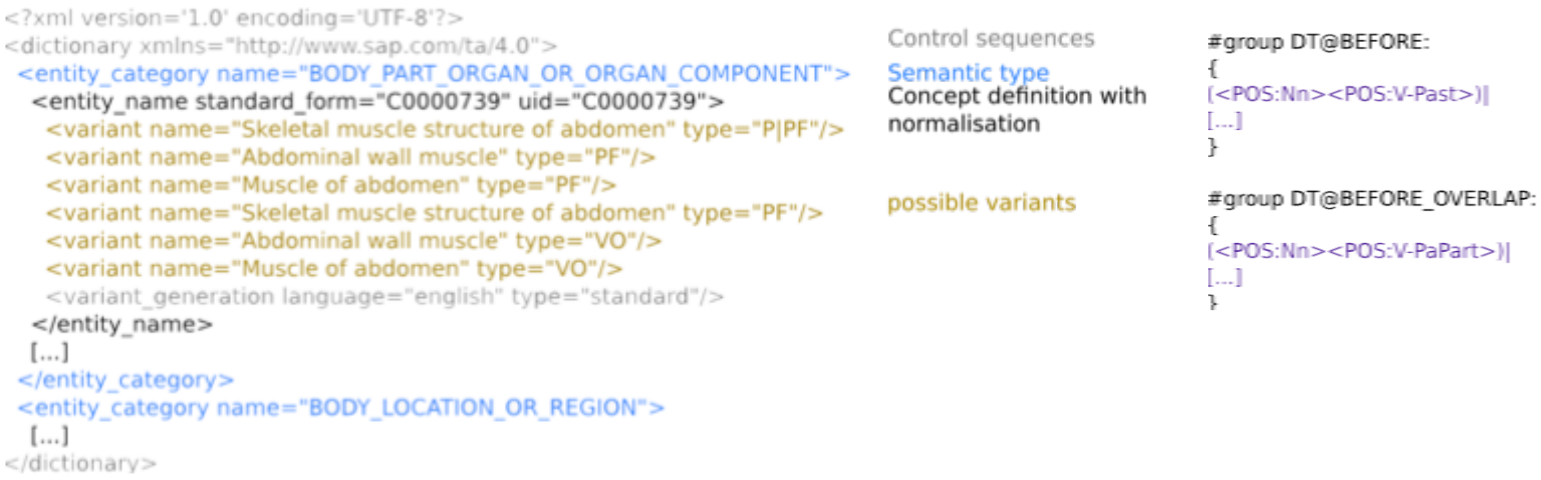
Reduction of Layers

Multi-Core and Parallelization

Text Retrieval and Extraction

IMDB Analysis Features for Applications – Textual Analysis of Medical Documents

1. Specify dictionary in XML and/or CGUL rules:



```
<?xml version='1.0' encoding='UTF-8'?>
<dictionary xmlns="http://www.sap.com/ta/4.0">
 <entity_category name="BODY_PART_ORGAN_OR_ORGAN_COMPONENT">
  <entity_name standard_form="C0000739" uid="C0000739">
   <variant name="Skeletal muscle structure of abdomen" type="P|PF"/>
   <variant name="Abdominal wall muscle" type="PF"/>
   <variant name="Muscle of abdomen" type="PF"/>
   <variant name="Skeletal muscle structure of abdomen" type="PF"/>
   <variant name="Abdominal wall muscle" type="VO"/>
   <variant name="Muscle of abdomen" type="VO"/>
   <variant_generation language="english" type="standard"/>
  </entity_name>
  [...]
 </entity_category>
 <entity_category name="BODY_LOCATION_OR_REGION">
  [...]
</dictionary>
```

Control sequences

Semantic type

Concept definition with normalisation

possible variants

```
#group DT@BEFORE:
{
(<POS:Nn><POS:V-Past>)|
[...]
}

#group DT@BEFORE_OVERLAP:
{
(<POS:Nn><POS:V-PaPart>)|
[...]
}
```

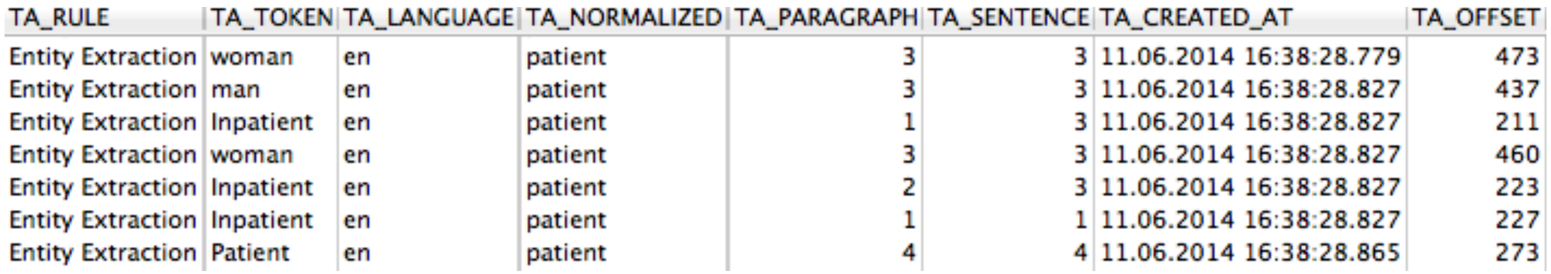
2. Compile XML dictionary for database and reference them in config file

IMDB Analysis Features for Applications – Textual Analysis of Medical Documents

3. Create fulltext index:

```
CREATE FULLTEXT INDEX "EXAMPLE"."EXAMPLE_INDEX" ON EXAMPLE"."EXAMPLE_DATA" ("TEXT")
CONFIGURATION 'PROJECT::MED_TERMS' ASYNC LANGUAGE DETECTION ('EN')
FUZZY SEARCH INDEX ON TEXT ANALYSIS ON TOKEN SEPARATORS '\/;,.:-_()[]<>!?*@+{}="&'
```

4. Get results in database table and use for further analyses:



TA_RULE	TA_TOKEN	TA_LANGUAGE	TA_NORMALIZED	TA_PARAGRAPH	TA_SENTENCE	TA_CREATED_AT	TA_OFFSET
Entity Extraction	woman	en	patient	3	3	11.06.2014 16:38:28.779	473
Entity Extraction	man	en	patient	3	3	11.06.2014 16:38:28.827	437
Entity Extraction	Inpatient	en	patient	1	3	11.06.2014 16:38:28.827	211
Entity Extraction	woman	en	patient	3	3	11.06.2014 16:38:28.827	460
Entity Extraction	Inpatient	en	patient	2	3	11.06.2014 16:38:28.827	223
Entity Extraction	Inpatient	en	patient	1	1	11.06.2014 16:38:28.827	227
Entity Extraction	Patient	en	patient	4	4	11.06.2014 16:38:28.865	273

IMDB Analysis Features for Applications – Statistical Analyses Functions

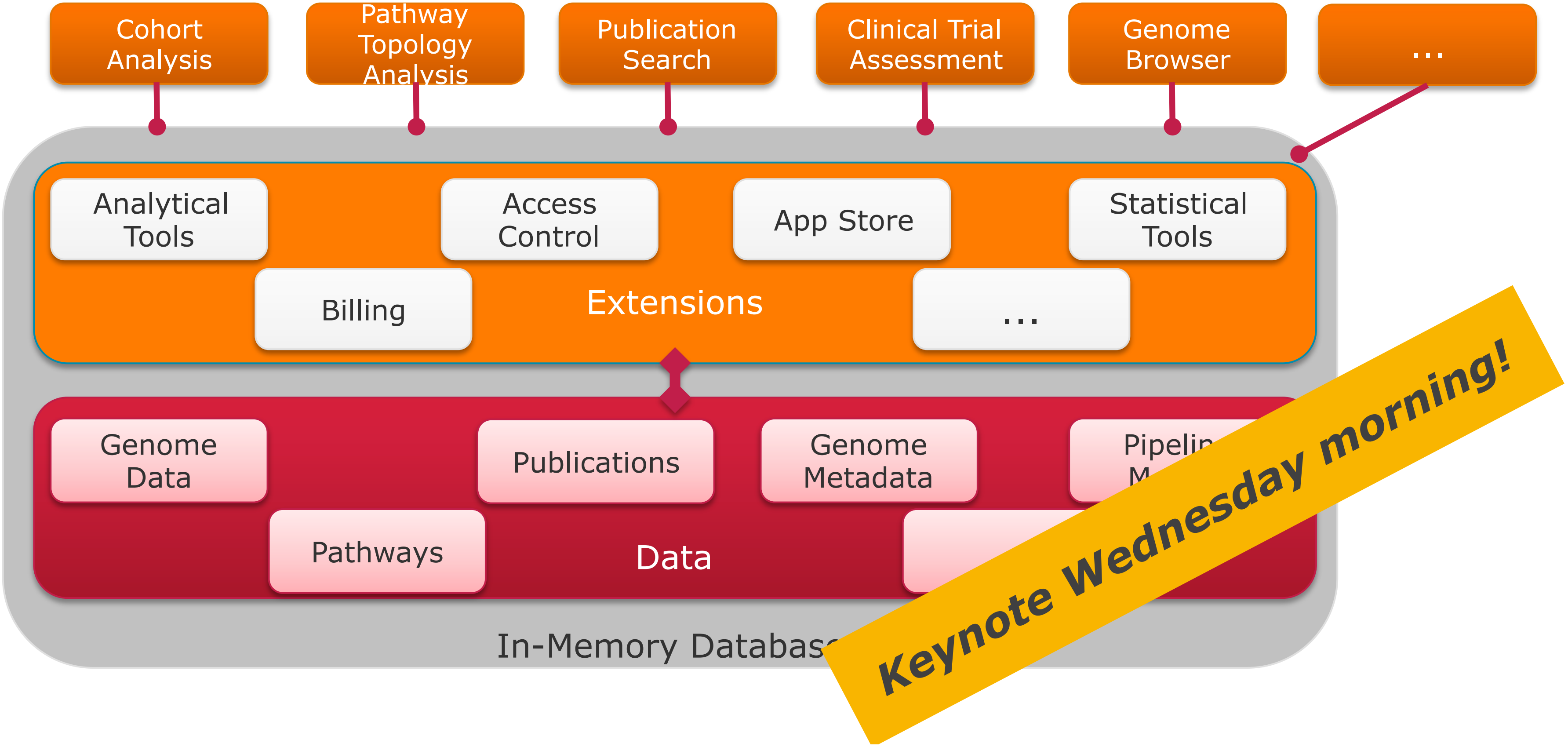
Reduction of Layers

Multi-Core and Parallelization

- IMDB provides specific **analysis functions** tightly integrated within the database, e.g. k-means or hierarchical clustering
- Highly parallelized and efficient using database framework
- Invoked as stored procedures via SQL statement:

```
CALL _SYS_AFL.PAL_HC(DATA_TAB, PARAM_TAB, COMBINEPROCESS_TAB, RESULT_TAB);
```

Analyze Genomes – An In-Memory Computing Platform



Keynote Wednesday morning!

Keep in contact with us.

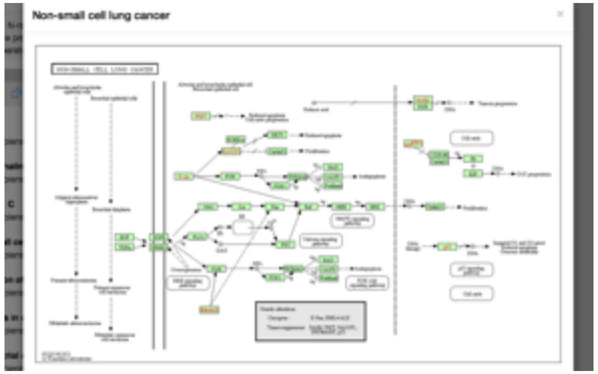
Cindy Fähnrich, M. Sc.
cindy.faehnrich@hpi.de

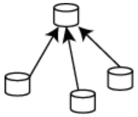
Dr. Matthieu-P. Schapranow
schapranow@hpi.de
http://we.analyzegenomes.com/

Hasso Plattner Institute
Enterprise Platform & Integration Concepts
August-Bebel-Str. 88
14482 Potsdam, Germany

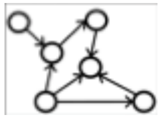
Backup/Further Questions

Medical Knowledge Cockpit for Clinicians
Pathway Topology Analysis





Unified access to multiple formerly disjoint data sources



Pathway analysis of genetic variants with graph engine

- Search in pathways is limited to “is a certain element contained” today
- Integrated >1,5k pathways from international sources, e.g. KEGG, HumanCyc, and WikiPathways, into HANA
- Implemented graph-based topology exploration and ranking based on patient specifics
- Enables interactive identification of possible dysfunctions affecting the course of a therapy before its start