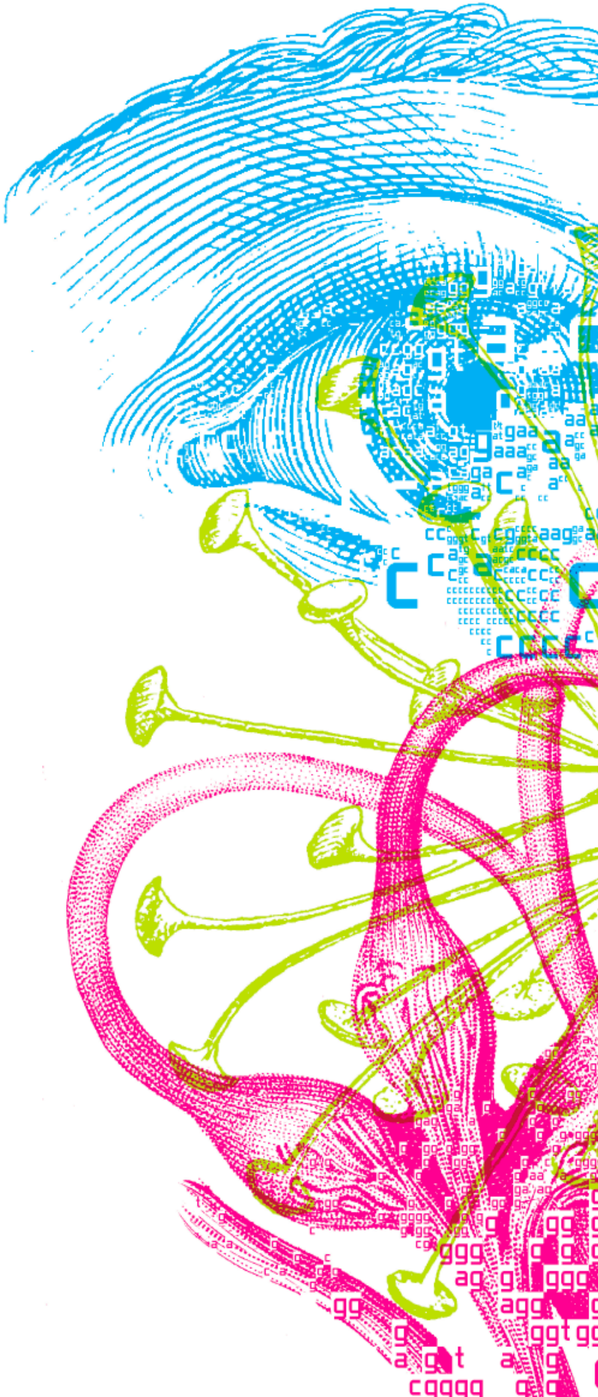
Netherlands
Bioinformatics
Centre

Pipelines & Workflows: Technical Discussion

Dr. Konstantinos Karasavvas

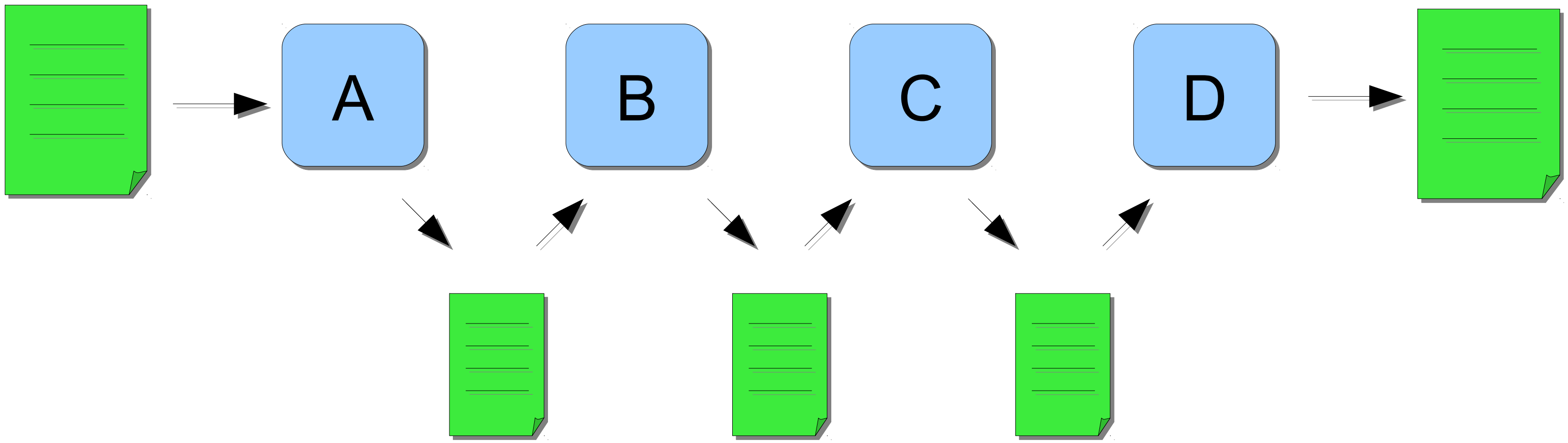
Leiden University Medical Center



Introduction

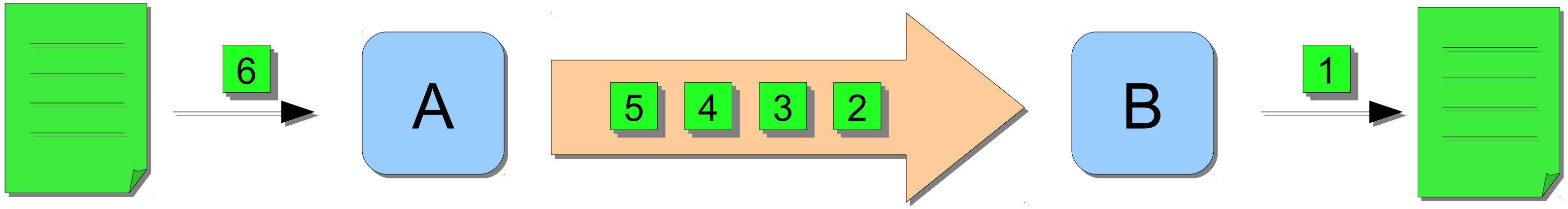
- Pipelines in Biology
    - extensively used to automate experiments
    - simple implementations
    - however, difficult to improve
        - biology is a complex domain
        - majority of legacy tools
            - build just for the job
            - bigger picture?
- Really quick overview
    - Pipelines & Streaming
    - Parallelisation
    - Workflows
        - Taverna
        - Galaxy

Sequential Execution: basic pipelines



- Coordinator
    - script
- Tasks operate on
    - complete files
- Example: DOS pseudo-pipelines
    - *Pipeline* implies streaming (and direction)

Streaming Pipelines: pipeline parallelism



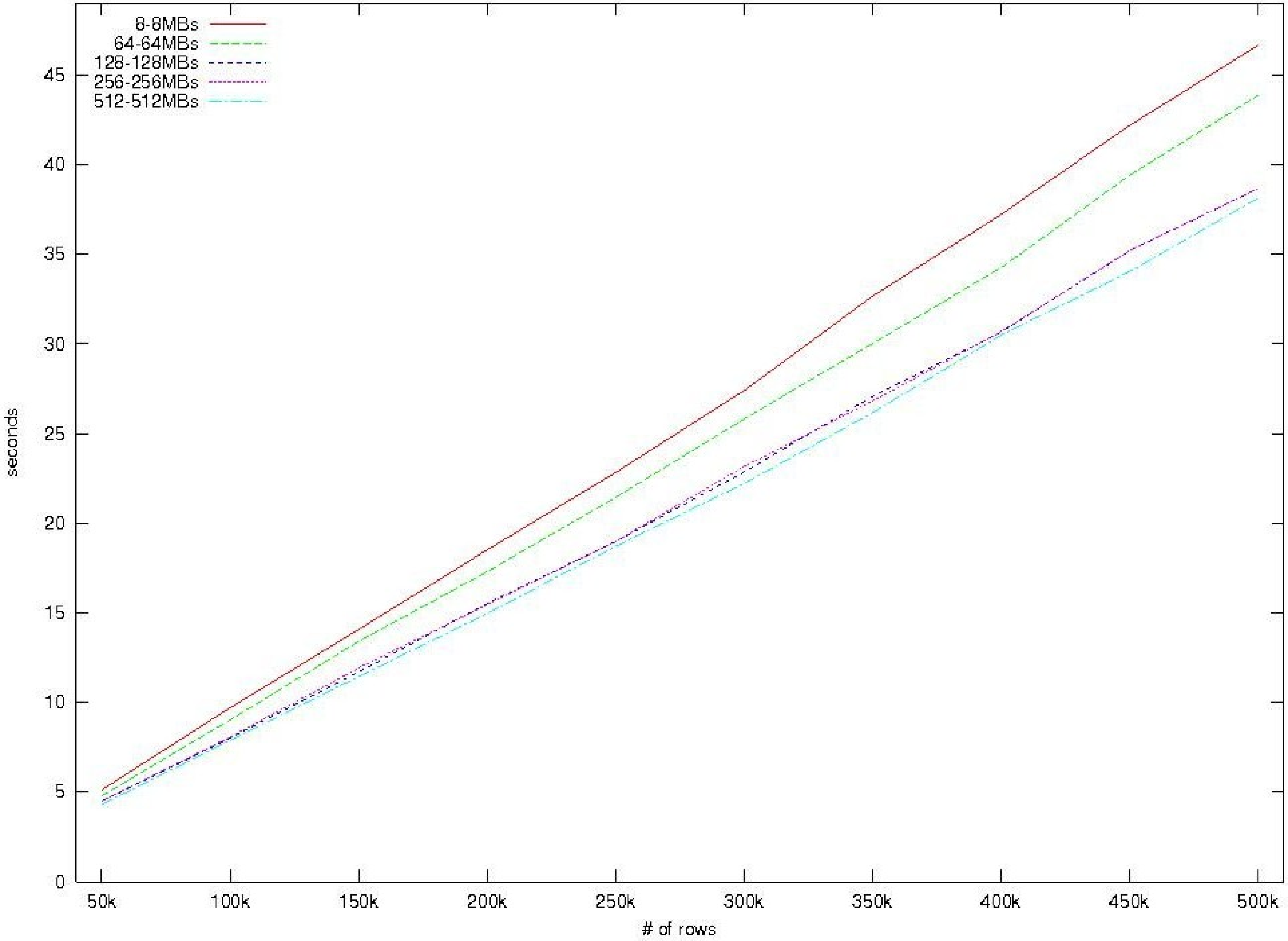
- Coordinator
  - pipeline 'engine'
- Tasks operate on
  - user-defined processable data chunks
- Example: Unix pipelines

Streaming Pipelines (1)

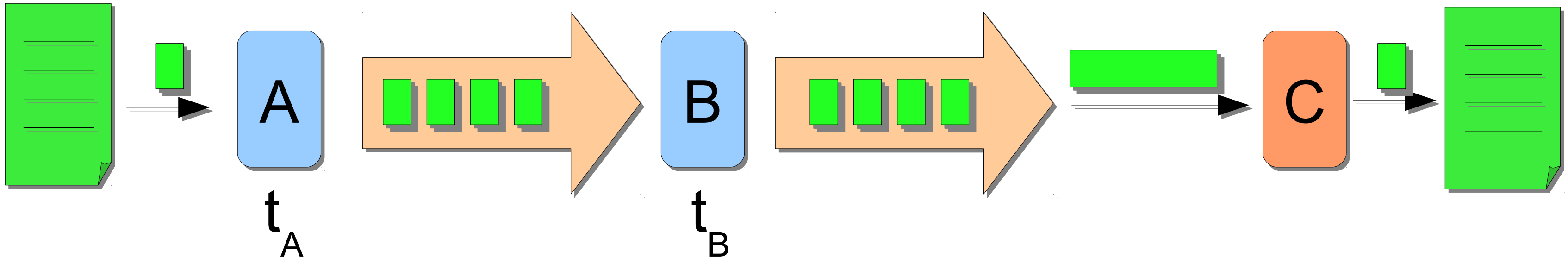
- A data chunk can be anything we want it to be
  - a delimited string (Unix)
  - an object (PowerShell, OGSA-DAI streaming engine)
  - bytes (multimedia frameworks)
    - different discussion
- Pipeline parallelism
  - data chunks are processed *concurrently*
  - depends on implementation!

Streaming Pipelines (2)

- Low memory footprint (old/cheap machines & cheaper virtualisation)

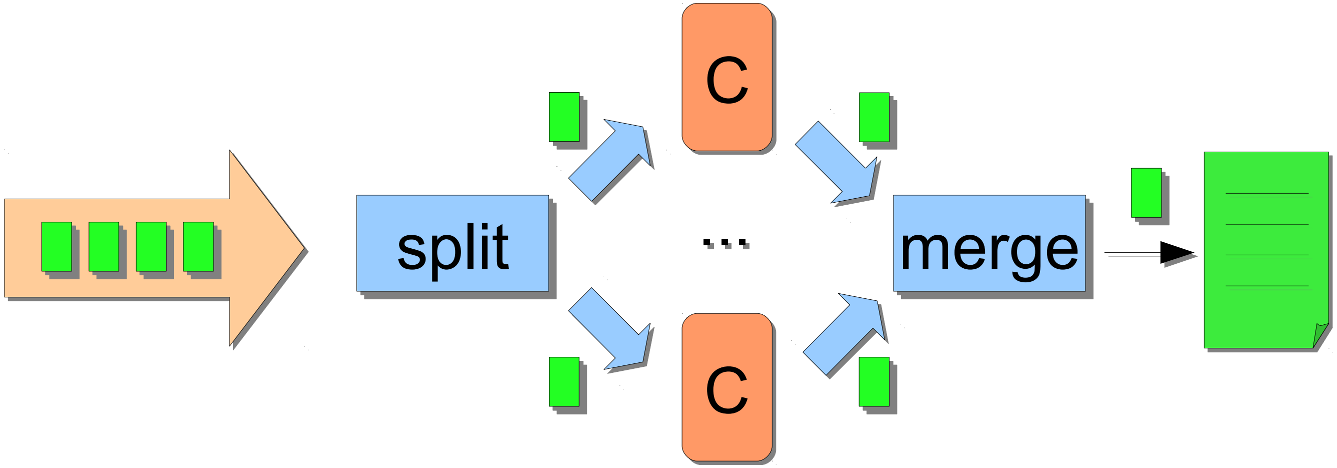


Pipelines: bottlenecks (1)



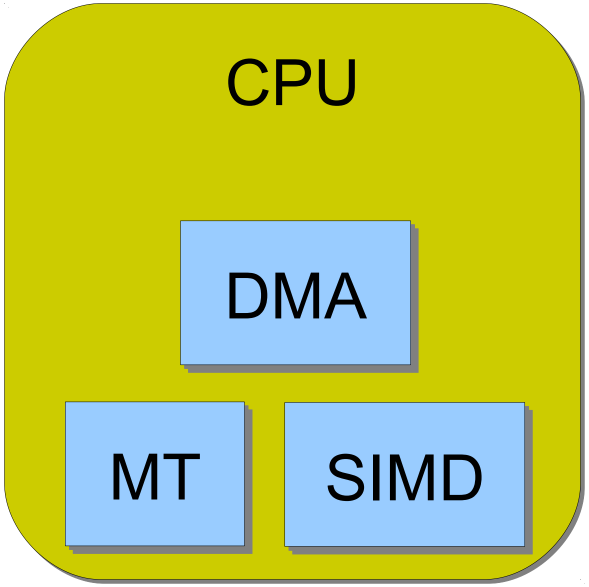
- granularity of a data chunk can be different
  - potential bottleneck: complete file is needed
- consistently different production/consumption rates

Pipelines: bottlenecks (2)



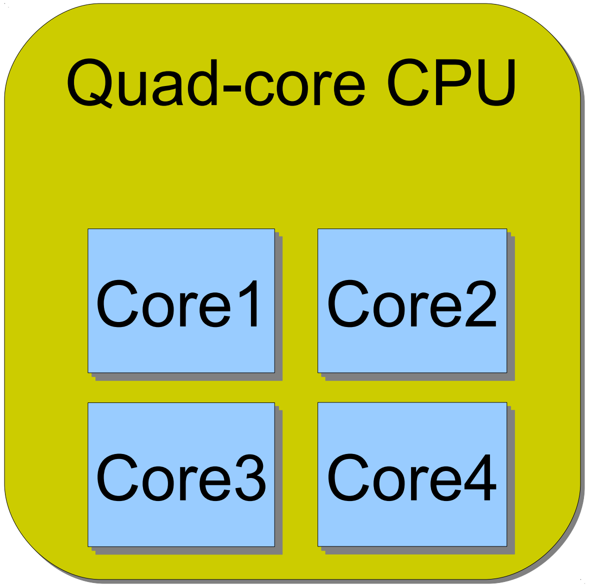
- resolution: parallelisation
  - really depends on the problem/algorithm, e.g. sort
- embarrassingly parallel
  - map/reduce
- other?
  - parallelize at the code level (sometimes that would require rewrite!)

Parallelisation (limited)



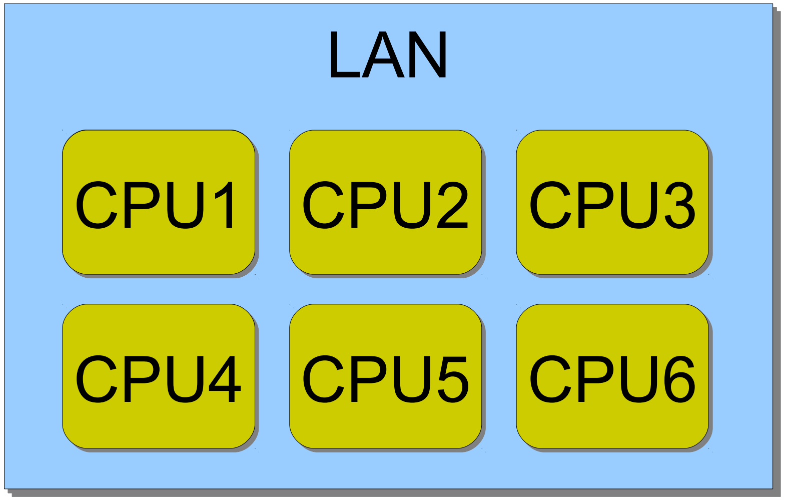
- single processor (one core)
  - DMA controller
    - e.g. channel I/O
  - hardware multi-threading
    - pseudo multi-core
    - e.g. cache misses
  - SIMD
    - e.g. array iterations
  - …
- software
  - dev.: threads, SIMD libs
- frequency scaling
  - power consumption issues

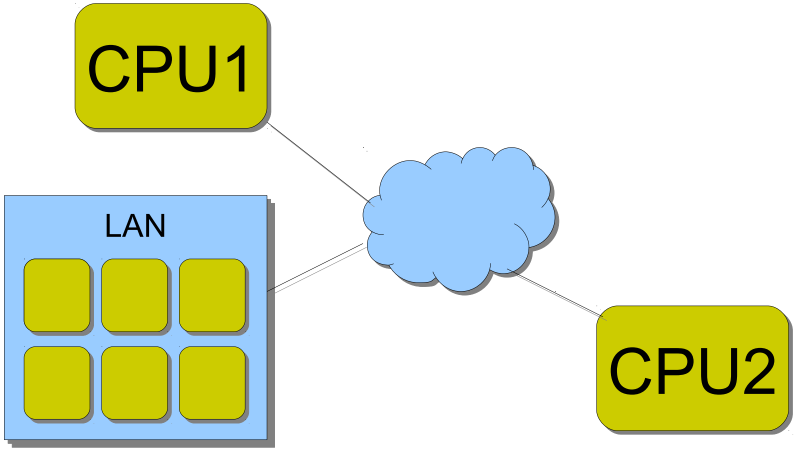
Parallelisation (1)



- multi-core processor
  - multi-processor machines
    - SMP, NUMA
  - shared memory
- performance shift
  - multi-core parallelism
  - > 2005
- software
  - dev.: threads
    - GUI + processing?
    - not enough
  - parallel programming models
    - e.g. OpenMP [, MPI]

Parallelisation (2)



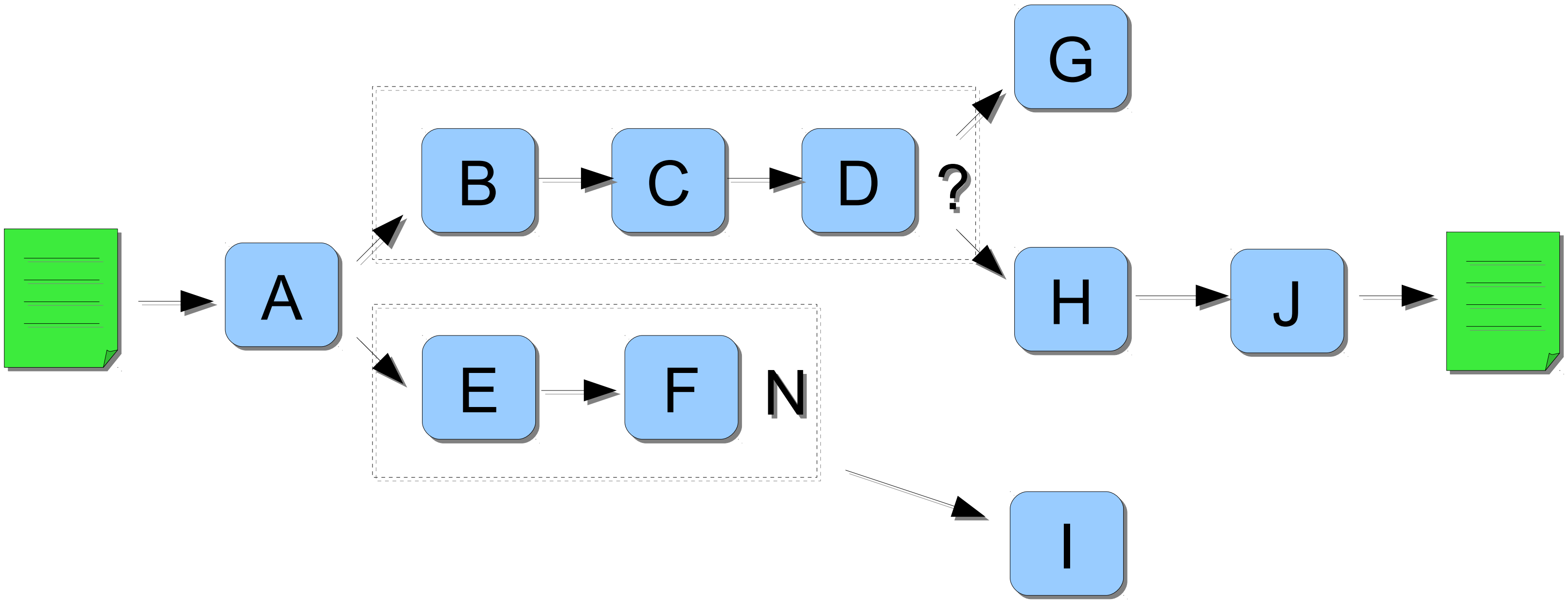


- multiple machines
  - clusters
    - usually symmetric
    - e.g. Beowolf cluster
  - grids
    - heterogeneous machines
    - loosely coupled
    - geographically dispersed
- distributed memory
  - data movement overhead!
- processing vs data movement?
- software
  - parallel programming models
    - e.g. MPI [, OpenMP]

Workflow Management Systems

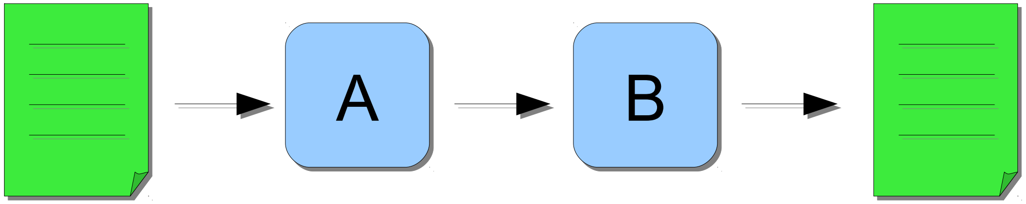
- components
  - high-level language
    - describes flow of work
    - expressive
  - enactment engine
    - play/pause/resume
    - provenance
  - editor
    - visualisation
- generic workflow systems
  - quite complicated

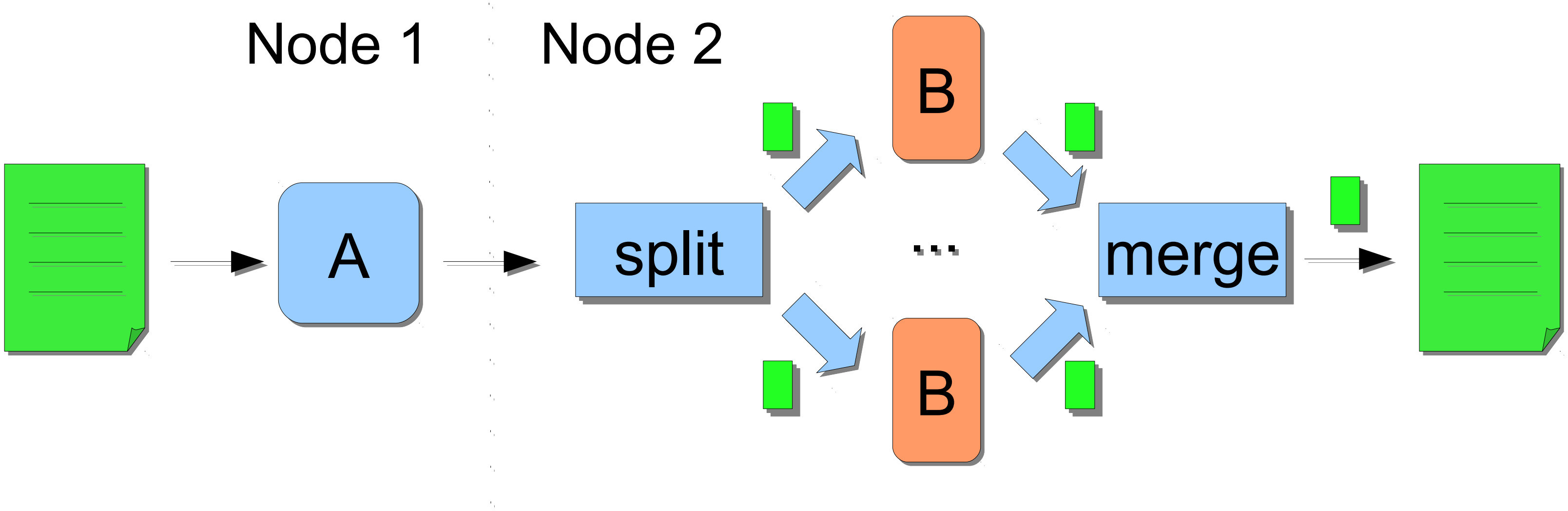
Workflow Languages: Example



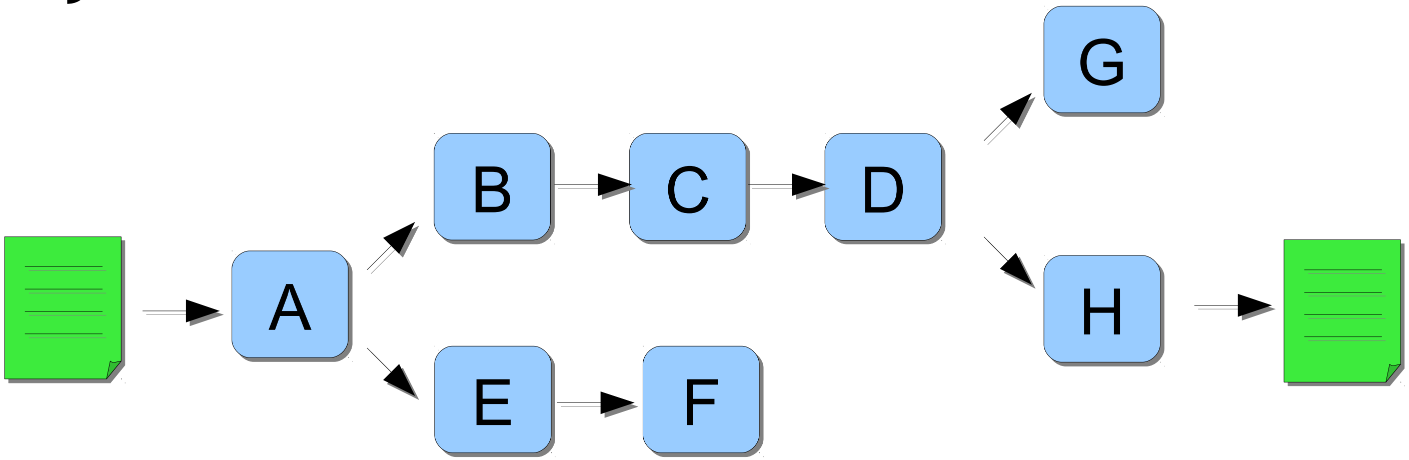
- parallelisation
- streaming (chunk processing)
- nesting
- conditionals and repetitions
  - many *workflow patterns* (http://www.workflowpatterns.com/)

Workflow Languages: Abstract vs Concrete



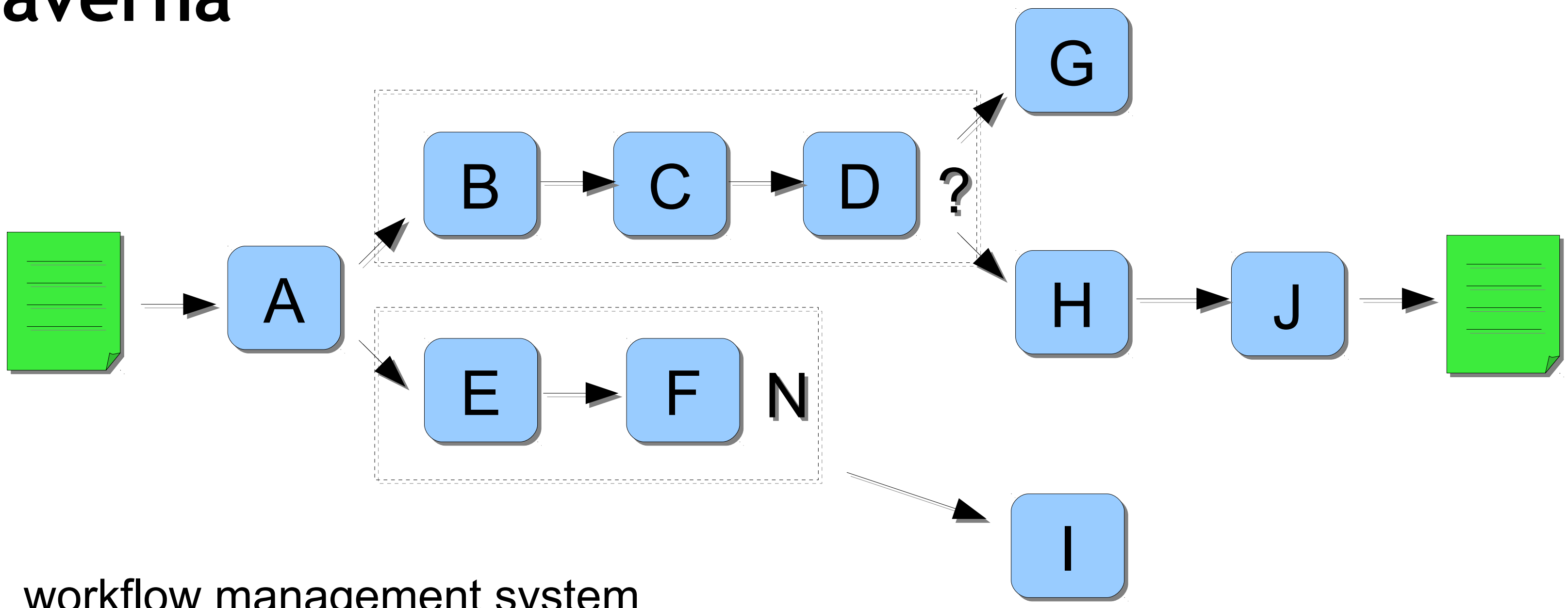


Galaxy

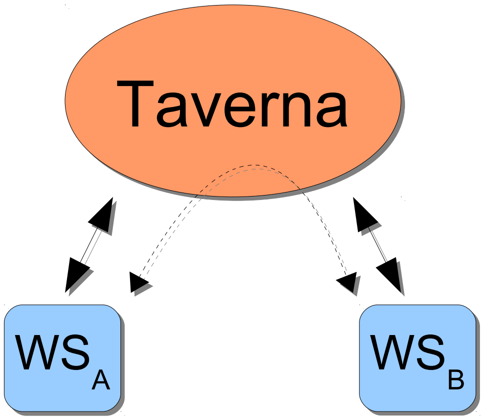


- started as a tool aggregation and data management system
    - simple pipelines
- components are (local) *black boxes* with file I/Os
- centralised system
    - tool distribution / administration issues
    - scaling via clustering (distributed memory)
    - tasks can be distributed per node
    - data movement
        - NFS

Taverna



- workflow management system
- components are mainly Web Services
  - beanshell scripts for small local tasks
  - but local/remote jobs/tools are supported
- naturally distributed (Bio-tools as WSs)
- data movement
  - main overhead
  - no streaming (WS- ?)



Workflow System Implementations

- Hundreds!
- Scientific Workflow Systems
  - data-intensive workflows
- Scientific Bioinformatics Workflow systems
  - wikipedia lists around 20
  - Galaxy, Taverna, Kepler, WS-VLAM, …

What we don't want!

- creating our own ad hoc pipeline/workflow implementation for every type of experiment
  - is that what is happening?
  - distribution of effort
  - no reuse

What we do want! (Ideally)

- a pipeline/workflow solution that we can re-use
  - effort can be concentrated from many groups
  - NBIC BioAssist?

What can we do?

- serious requirement capture of our pipelines
  - how important is expressing the problem?
  - how important is efficiency?
    - which parts can be streamed or parallelised?
    - which scaling solution covers our needs?
    - can we identify the tools that create bottlenecks?
      - could we just optimise those?
      - SIMD libs, OpenMP, MPI, … ?
- serious investigation into existing solutions
  - many Bioinformatics *Workflow* Management Systems
  - are they good enough?
  - should we concentrate efforts on extending one?

Discussion

Discussion

- How many groups work with *pipelines*?

Discussion

- How many groups work with *pipelines*?
- What pipeline requirement gathering efforts were made?
  - can we concentrate such efforts?

Discussion

- How many groups work with *pipelines*?
- What pipeline requirement gathering efforts were made?
  - can we concentrate such efforts?
- What workflow system investigation were made?
  - can we concentrate such efforts?

Discussion

- How many groups work with *pipelines*?
- What pipeline requirement gathering efforts were made?
  - can we concentrate such efforts?
- What workflow system investigation were made?
  - can we concentrate such efforts?
  - did we select Galaxy?
    - do we know why?
      - easy to use? (for whom?)
      - popular?

Discussion

- How many groups work with *pipelines*?
- What pipeline requirement gathering efforts were made?
  - can we concentrate such efforts?
- What workflow system investigation were made?
  - can we concentrate such efforts?
  - did we select Galaxy?
    - do we know why?
      - easy to use? (for whom?)
      - popular?
    - can we make a list of issues with Galaxy?
      - suggest them to the Galaxy team
      - implement them ourselves

Discussion

- How many groups work with *pipelines*?
- What pipeline requirement gathering efforts were made?
  - can we concentrate such efforts?
- What workflow system investigation were made?
  - can we concentrate such efforts?
  - did we select Galaxy?
    - do we know why?
      - easy to use? (for whom?)
      - popular?
    - can we make a list of issues with Galaxy?
      - suggest them to the Galaxy team
      - implement them ourselves
  - common CPU-/RAM-intensive tools that we could optimise?