COBRA.jl

Accelerating Systems Biomedicine

JuliaCon 2017

Laurent Heirendt, Ph.D.
@laurentheirendt - June 23rd, 2017

Luxembourg Centre for Systems Biomedicine

Fonds National de la Recherche Luxembourg

U.S. DEPARTMENT OF ENERGY

PARKINSON National Centre of Excellence in Research

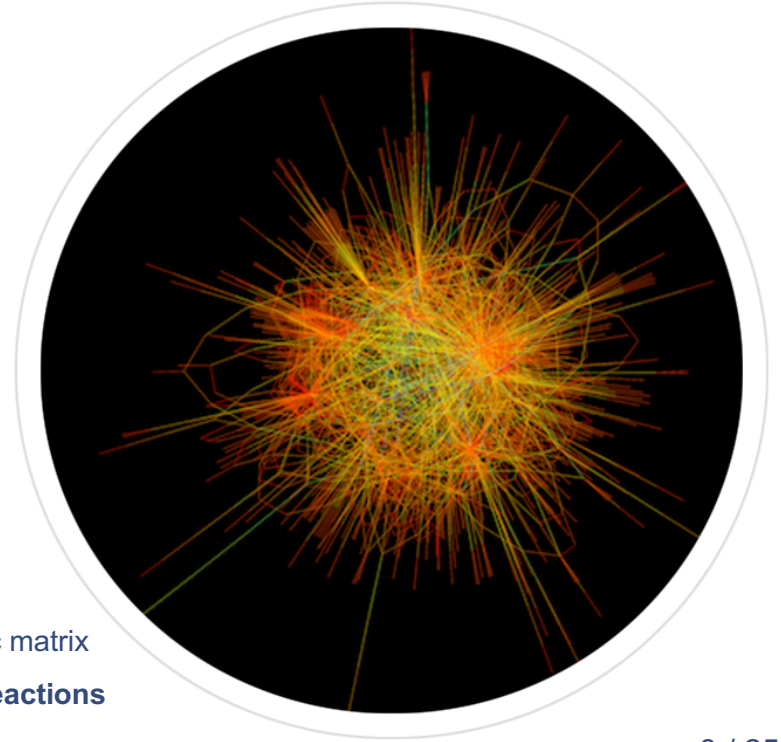
UNIVERSITÉ DU LUXEMBOURG

Outline

1. **CO**nstraint-**b**ased **R**econstruction and **A**nalysis (COBRA)
2. COBRA & Julia: large- and huge-scale modelling
3. Flux balance and flux variability analysis (FBA & FVA)
4. **distributedFBA.jl**, part of COBRA.jl
5. Benchmarking
6. Short how-to guide
7. Conclusions & Outlook

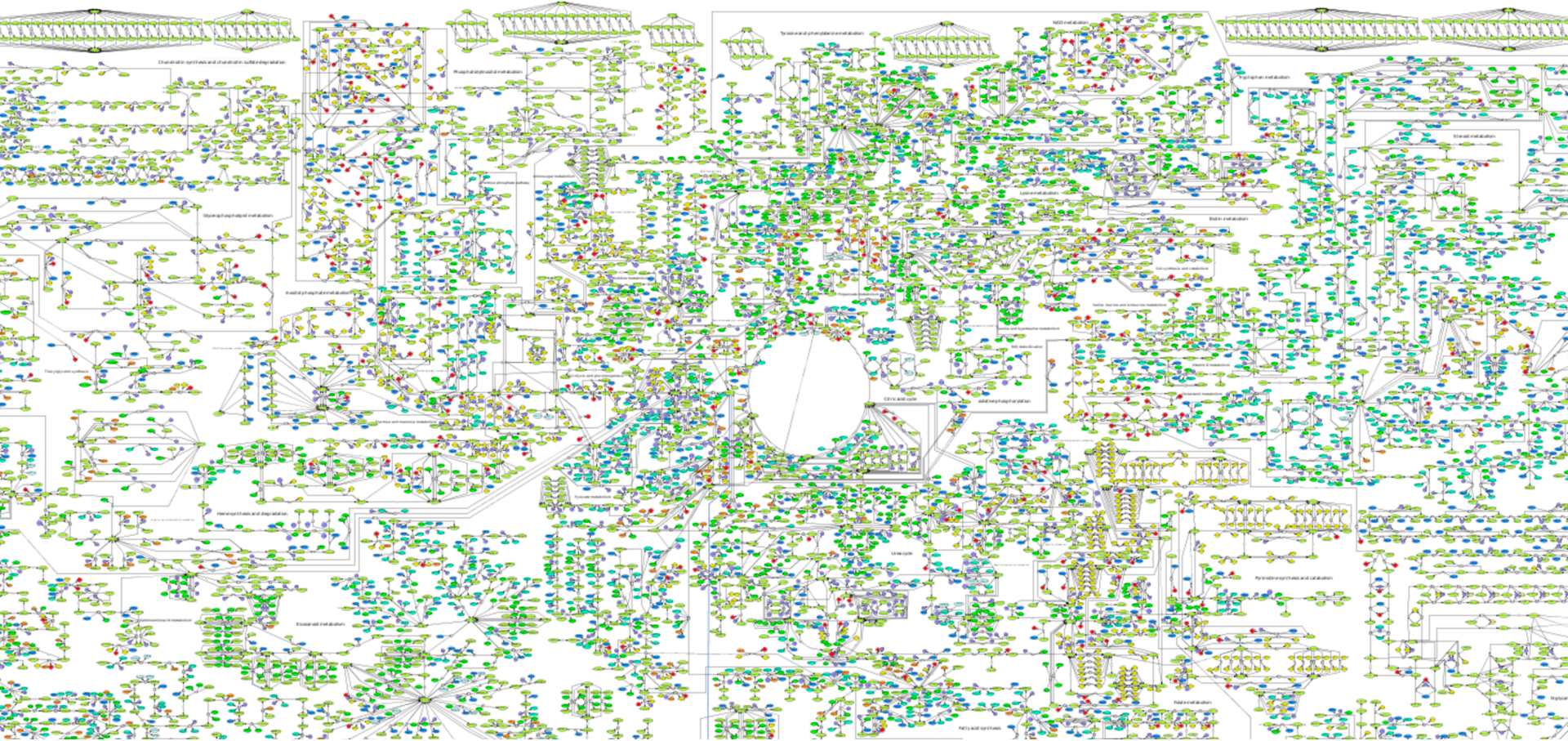
What is COBRA?

- **COBRA** - **CO**nstraint-**b**ased **R**econstruction and **A**nalysis
- Widely used approach for
  - modelling genome-scale **biochemical networks**
  - performing integrative analysis of omics data in a network context.
- COBRA has developed rapidly in recent years



Representation of a stoichiometric matrix
with **2785 metabolites** and **3820 reactions**
(Human model Recon 1)

http://vmh.life



The stoichiometric matrix

- Generally, a chemical equation is written as:

$$\underbrace{aA + bB}_{\text{reactants}} \overset{v}{\rightarrow} \underbrace{cC + dD}_{\text{products}}$$

- a, b, c, d are *stoichiometric coefficients*
- v is the *reaction rate* or *metabolic flux* (generally unknown)
- Steady-state mass balance: $Sv = 0$, with S being the stoichiometric matrix with m metabolites and n reactions

$$S := \begin{bmatrix} -a \\ -b \\ c \\ d \end{bmatrix}$$

- In this case, S is a 4×1 matrix (4 metabolites participate in 1 biochemical reaction)

Why COBRA?

- We do not possess sufficiently detailed parameter data to precisely model an organism at genome-scale (in the biophysical sense)
- COBRA methods may not provide a unique solution, but provide a reduced set

  ⟶ guide **biological hypothesis development**

- All COBRA predictions are derived from optimization problems of the form:

$$\begin{aligned} \min_{v \in \mathbb{R}^n} \quad & \psi(v) \\ \text{s.t.} \quad & Sv = b \\ & Cv \leq d \\ & l \leq v \leq u \end{aligned}$$

n, m	number of reactions, metabolites
$v \in \mathbb{R}^n$	rate of each biochemical reaction
$\psi : \mathbb{R}^n \to \mathbb{R}$	lower semi-continuous, convex function
$S \in \mathbb{R}^{m \times n}$	stoichiometric matrix
b	vector of known metabolic exchanges
C, d	additional linear inequalities
u, l	upper, lower bounds of reaction rates

Flux balance analysis (FBA)

Goal: determine a steady-state reaction rate of one biochemical reaction based on mass balance (input = output)

Steady-state: choosing a coefficient vector $c \in \mathbb{R}^n$ and letting $\psi(v) := c^T v$ and $b := 0$

FBA is equivalent to solving the linear program (LP):

$$\begin{aligned} \min/\max_{v \in \mathbb{R}^n} \quad & c^T v \\ \text{s.t.} \quad & Sv = 0 \\ & l \leq v \leq u, \end{aligned}$$

which yields a unique objective $c^T v^\star$, but multiple alternate optimal solutions $v^\star$ may exist.

Flux variability analysis (FVA)

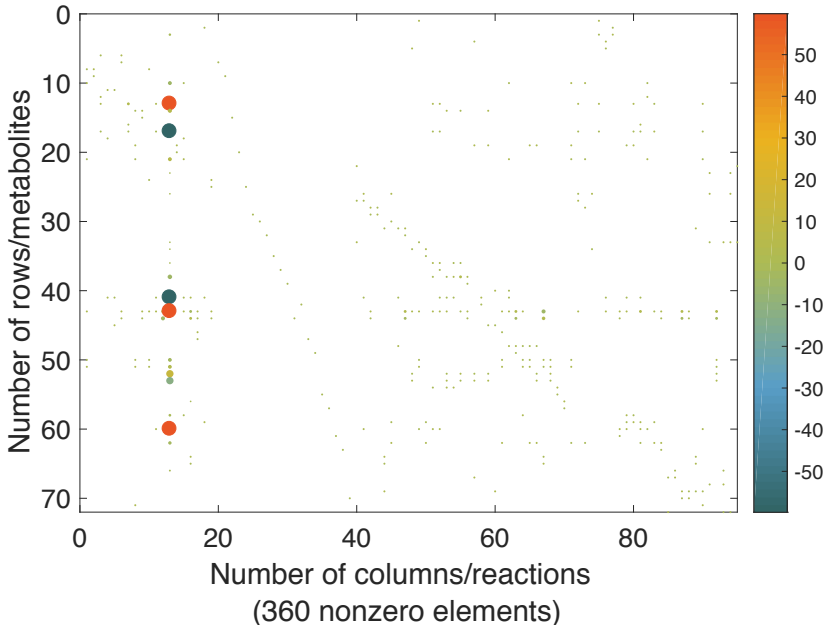
Challenge: the biologically correct coefficient vector $c \in \mathbb{R}^n$ is usually **not known**.

Exploration of the set of steady states relies on running FBA for many $c \in \mathbb{R}^n$

- $2n$ linear optimization problems
- embarrassingly parallel problem

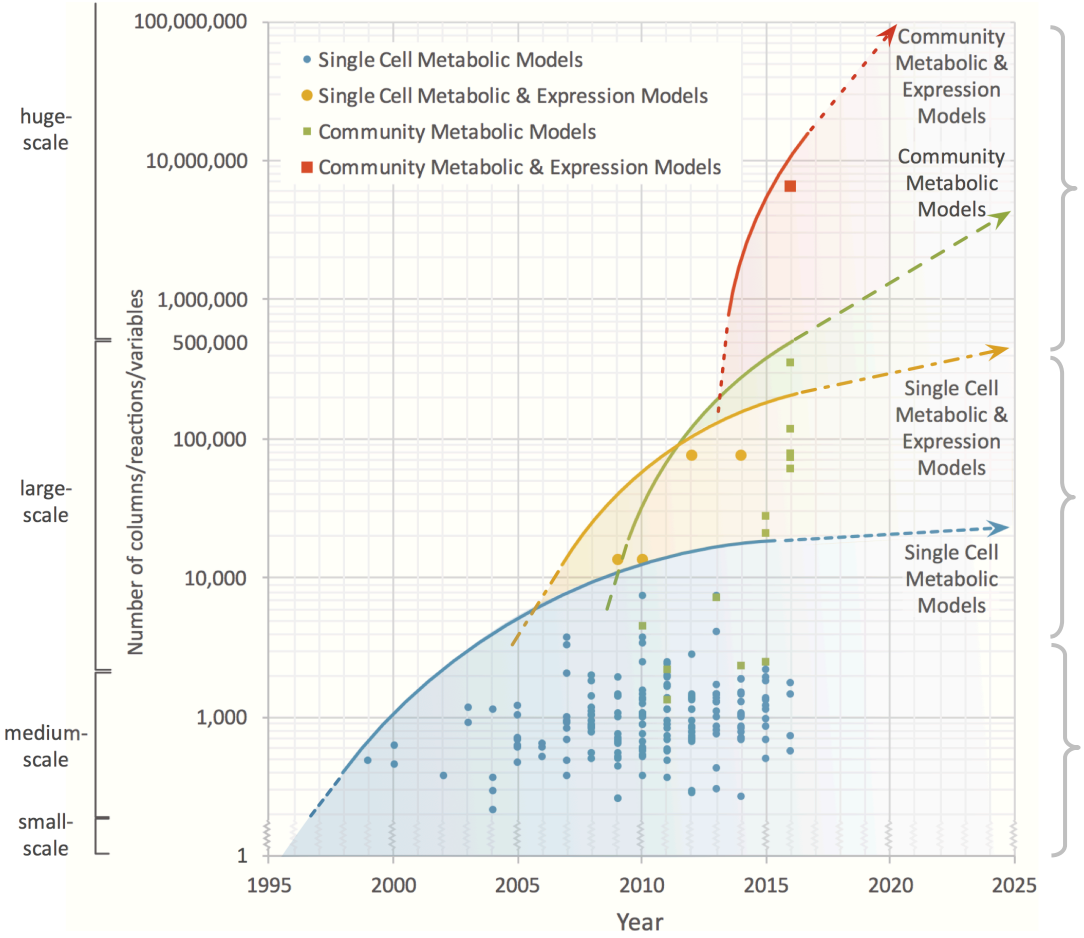
Determine the extremes for each reaction rate by:

- choosing a coefficient vector $d \in \mathbb{R}^n$
  with 1 non-zero entry
- minimizing/maximizing $\psi(v) := d^T v$
  s.t. the additional constraint $d^T v \geq \gamma \cdot c^T v^*$
  $(\gamma \in]0, 1[)$



E.*coli* core model (95 reactions, 75 metabolites)

COBRA & Julia: large- and huge-scale modelling (1/2)

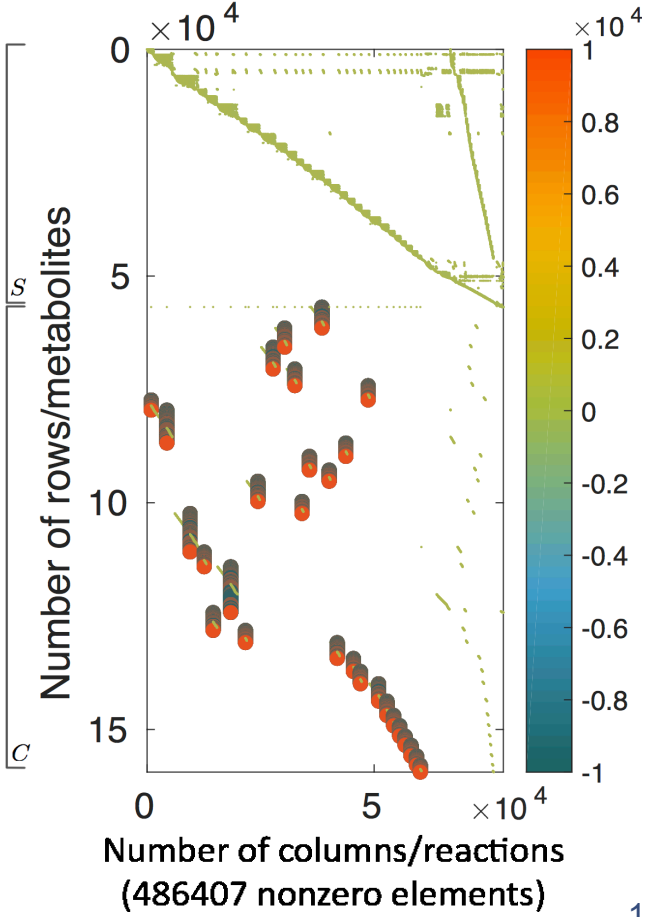


COBRA & Julia: large- and huge-scale modelling (2/2)

- For kilo-scale models (n ~ 1000), FVA can be performed efficiently using existing methods:
  - FVA (The COBRA Toolbox)
  - fastFVA (The COBRA Toolbox) MEX - C
  - COBRApy implementation

- Existing implementations perform best when using only 1 computing node with a few cores

→ **temporal limiting factor** when exploring the steady state solution space of large- or huge-scale models.



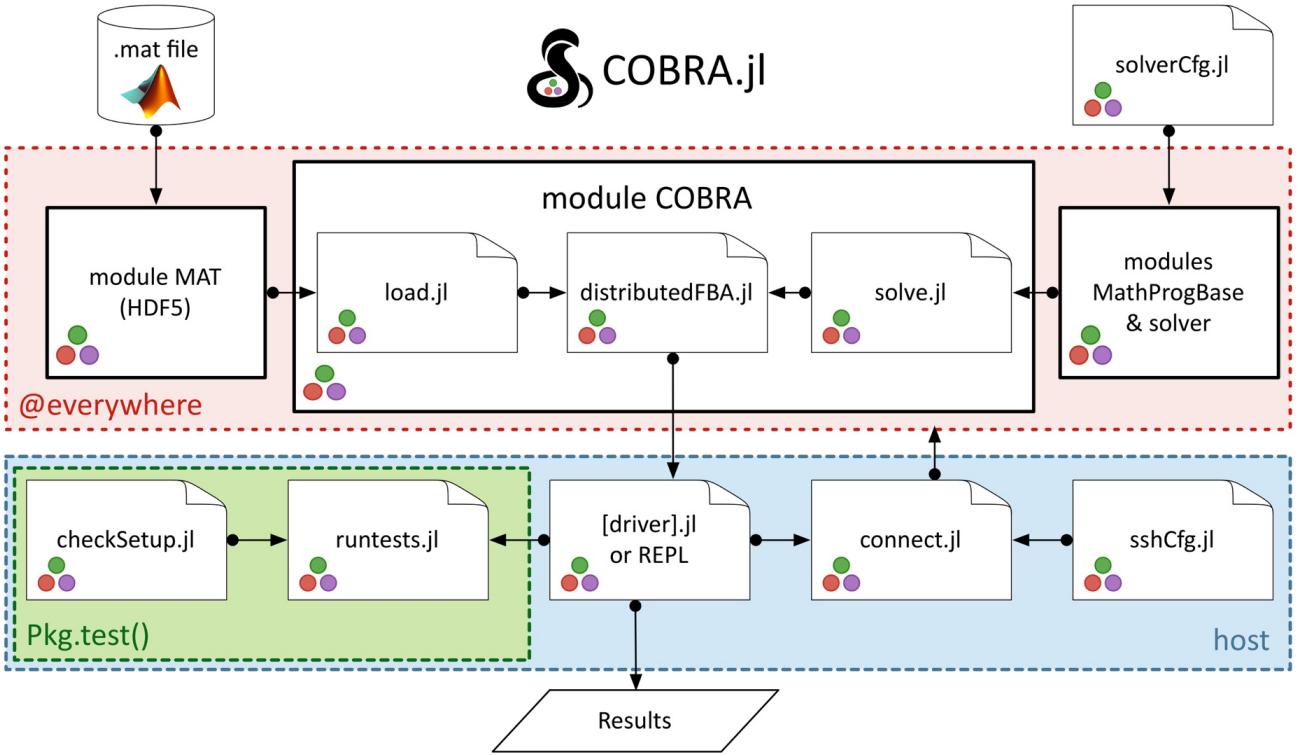
github.com/opencobra/COBRA.jl

- ✓ High-level, high-performance code
- ✓ High-memory multi-nodal analysis
- ✓ Registered package
- ✓ Well documented, maintained and tested package build passing build passing
- ✓ High coverage codecov 96%
- ✓ Tutorials (interactive notebooks) jupyter

DistributedFBA.jl - Overview

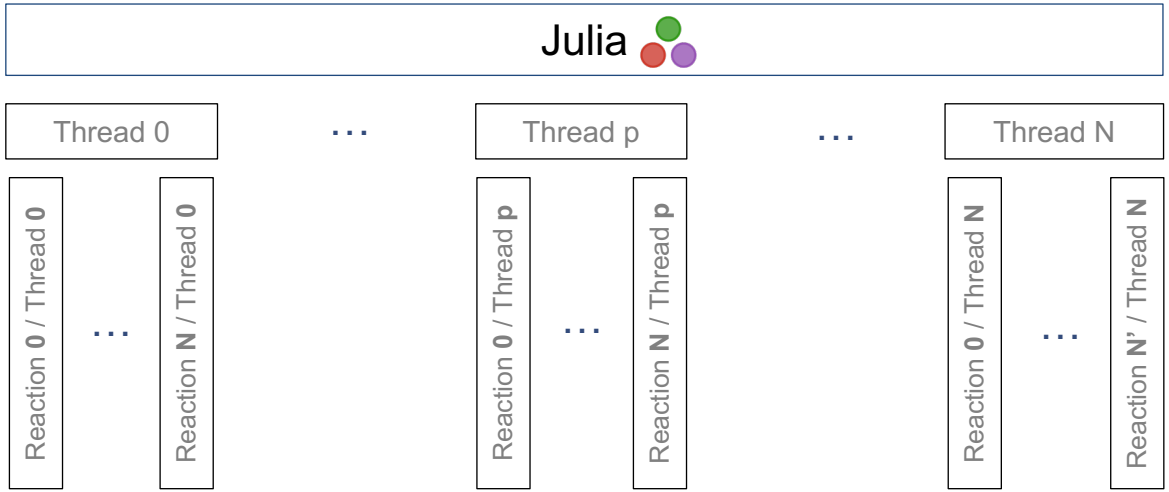
Input: a .mat (HDF5) file with data of a COBRA model (structure)

Output: Minimum/maximum reaction rates for each reaction and corresponding flux vectors



DistributedFBA.jl – Distribution Mechanism

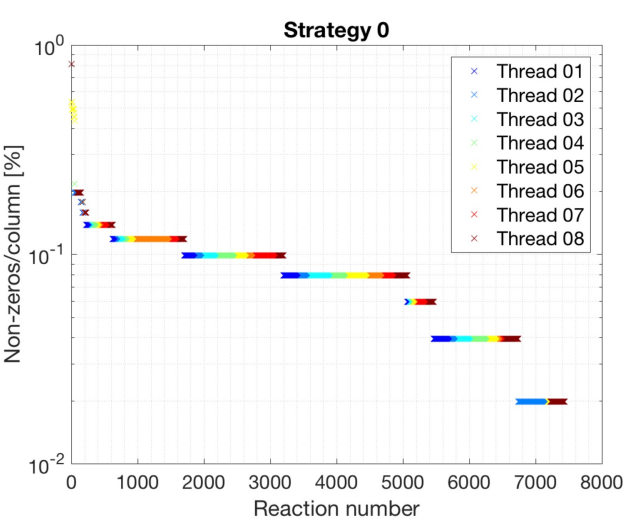
Distribution of blocks of reactions to threads (workers):

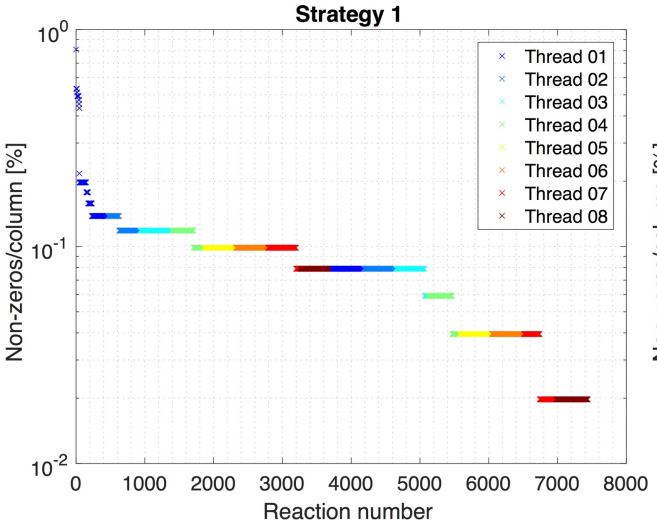


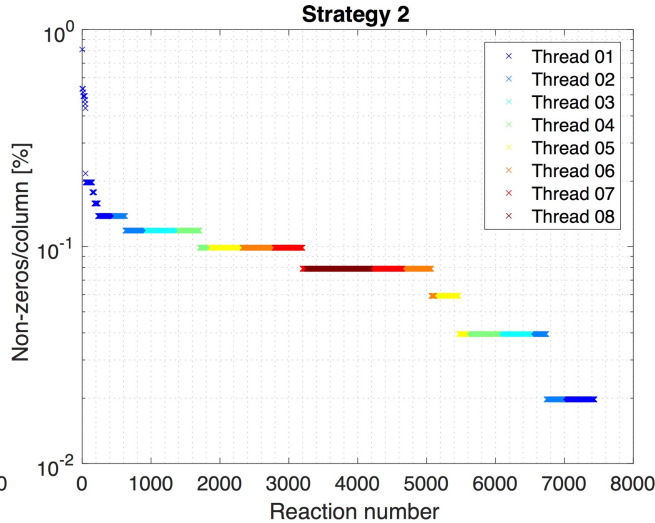

```
# distribution across workers
@sync for (p, pid) in enumerate(workers())
    for iRound = 0:1
        @async R[p, iRound + 1] = @spawnat (p + 1) begin
            m = buildCobraLP(model, solver)
            loopFBA(m, rxnsList[rxnsKey[p]], ...)
        end
    end
end
```

DistributedFBA.jl – Distribution strategies

- Static distribution strategies:
  - $s = 0$: Blind splitting: default random distribution
  - $s = 1$: Extremal dense-and-sparse splitting
  - $s = 2$: Central dense-and-sparse splitting







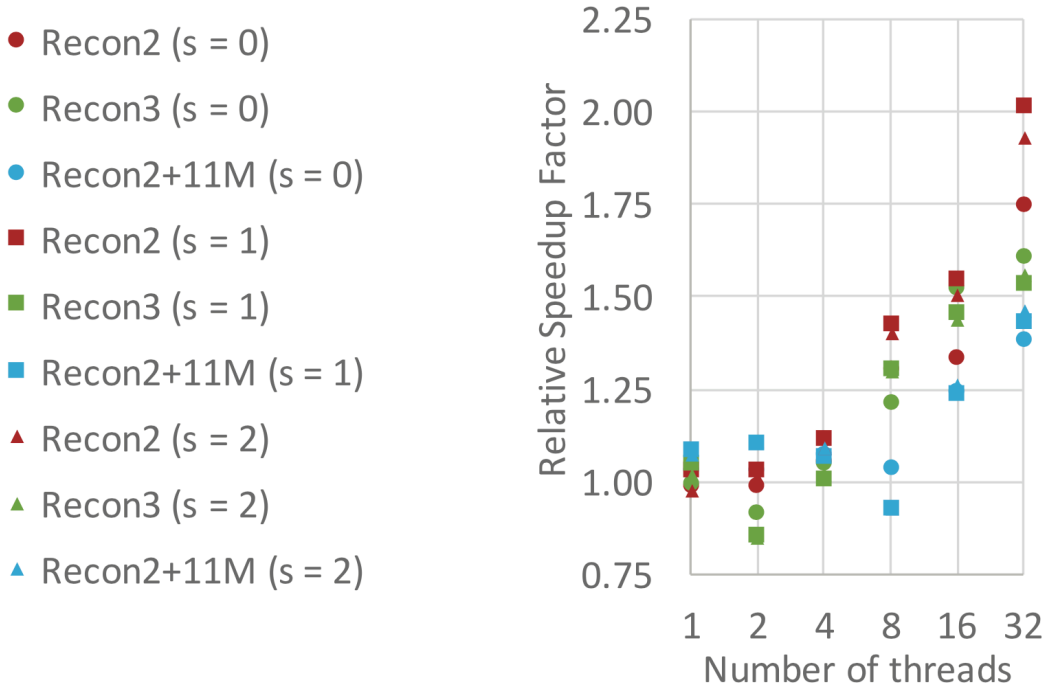
- Dynamic distribution strategies may also be implemented

DistributedFBA.jl – Performance and Benchmarks

#	Model name	Metabolites m	Reactions n	References
1	Recon1	2785	3820	Duarte *et al.* (2007)
2	Recon2	5063	7440	Thiele *et al.* (2013)
3	Recon3	7866	12 566	
4	Recon2 + 11M	19 714	28 199	Heinken *et al.* (2015)
5	Multi-organ	47 123	61 230	
6	SRS064645	89 756	99 104	Magnusdottir *et al.* (2016)
7	SRS011061	126 682	139 420	Magnusdottir *et al.* (2016)
8	SRS012273	186 662	208 714	Magnusdottir *et al.* (2016)

- Performance comparisons:
  - relative speedup to *fastFVA* [1]
  - distribution strategies
  - theoretical predictions – Amdahl's Law

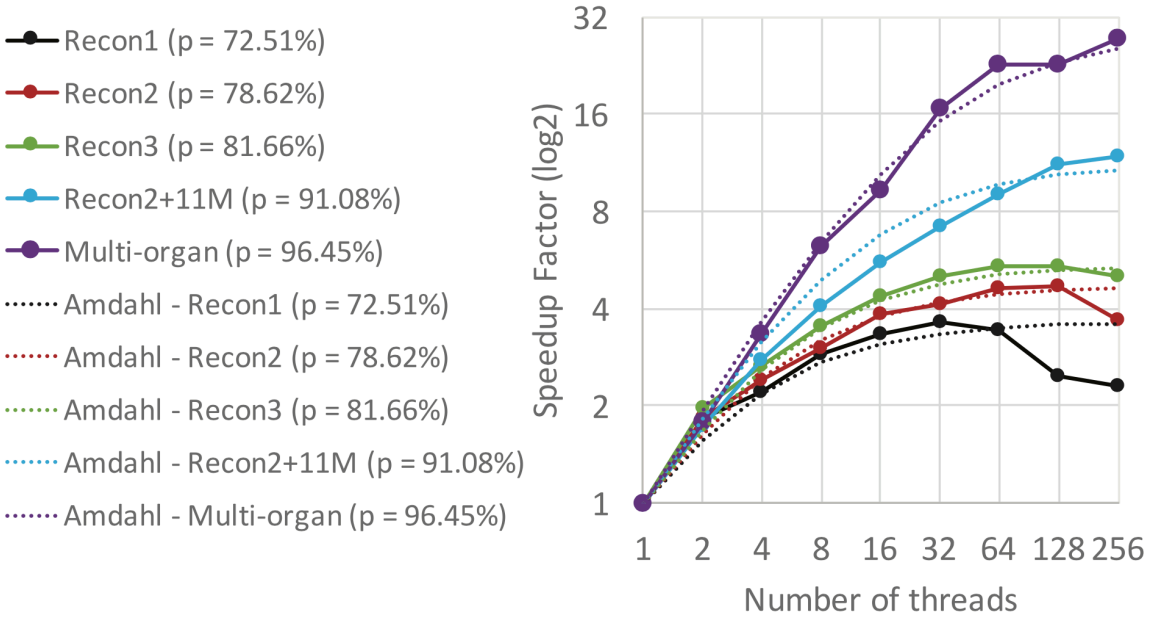
DistributedFBA.jl – Benchmarks



Uninodal speedup factor relative to *fastFVA* as a function of threads and distribution strategy s.

DistributedFBA.jl – Scalability

- Theoretical speedup factor given by Amdahl's law $\left(1 - p + \frac{p}{N}\right)^{-1}$ with N threads.
- The larger the model, the higher the parallelizable fraction p



Multi-nodal speedup in latency and Amdahl's law $(s = 0)$

Short how-to guide (1/4)

- Changing the COBRA solver

```
using COBRA
```

```
# change the COBRA solver
solver = changeCobraSolver("CPLEX"); # any solver supported by MathProgBase.jl
```

- Load an existing COBRA model (using MAT.jl)

```
# load the stoichiometric matrix S from a struct named model in the specified .mat file
model = loadModel("ecoli_core_model.mat", "S", "model");
```

Short how-to guide (2/4)

- Perform flux balance analysis (FBA)

```
# set the reaction list (only one reaction)
rxnsList = 13

# select the reaction optimization mode
#  0: only minimization
#  1: only maximization
#  2: maximization and minimization
rxnsOptMode = 1

~, maxFlux = distributedFBA(model, solver, nWorkers=1, rxnsList=rxnsList, rxnsOptMode=rxnsOptMode, ...);
```

Short how-to guide (3/4)

Perform flux variability analysis (FVA)

- Initialize the workers

```
include("$(Pkg.dir("COBRA"))/src/connect.jl")

# specify the total number of parallel workers
nWorkers = 512

# create a parallel pool
workersPool, nWorkers = createPool(nWorkers)

@everywhere using COBRA
```

- Run flux variability analysis

```
# launch the distributedFBA process with all reactions
minFlux, maxFlux, optSol, fbaSol, fvamin, fvamax  = distributedFBA(model, solver, nWorkers=nWorkers, ...)
```

Short how-to guide (4/4)

- Flux balance analysis of distinct reactions

```
rxnsList = [1; 18; 10; 20:30; 90; 93; 95]
rxnsOptMode = [0; 1; 2; 2+zeros(Int, length(20:30)); 2; 1; 0]

# run only a few reactions with rxnsOptMode and rxnsList
minFlux, maxFlux = distributedFBA(model, solver, nWorkers=4,  rxnsList=rxnsList, rxnsOptMode=rxnsOptMode, ...);
```

- Save results

```
saveDistributedFBA("results.mat")
```

Conclusions & Outlook (1/2)

- DistributedFBA.jl outperforms other implementations for large-scale models:

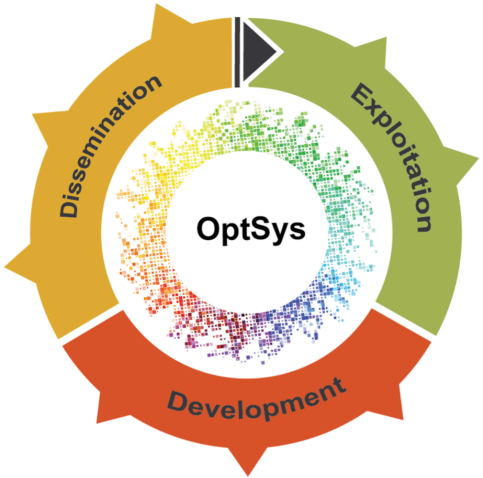
  - ✓ Scalability matches theoretical predictions
  - ✓ Resources are optimally used
  - ✓ Open-source
  - ✓ Platform independent
  - ✓ No node/thread limitations

→

- Timely analysis of large and huge-scale biochemical networks
- Analysis possibilities in the COBRA community lifted to another level

Conclusions & Outlook (2/2)

OptSys project

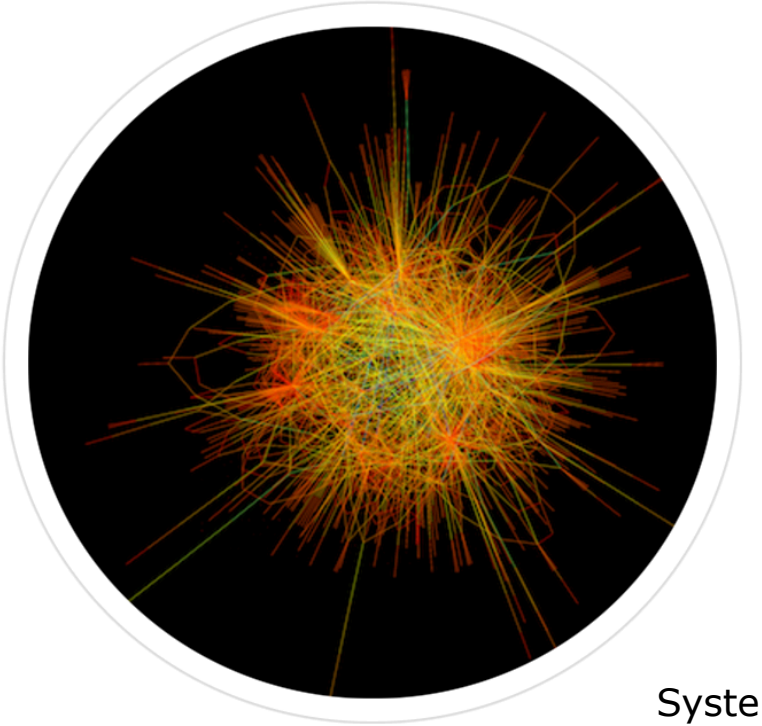


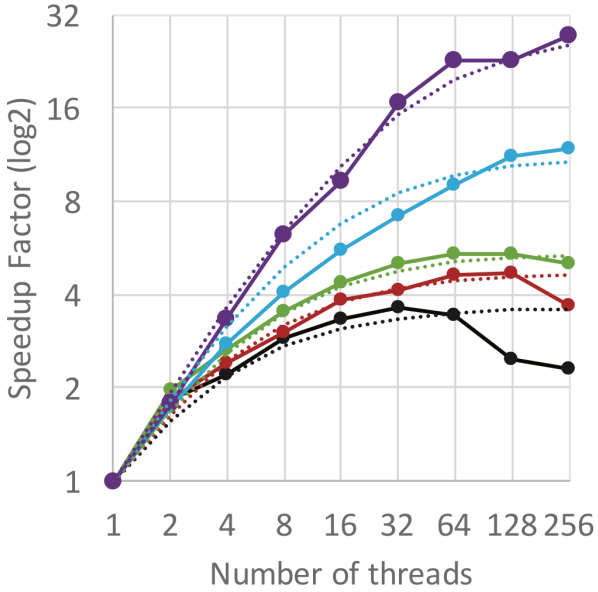
- Run distributedFBA.jl on COBRA models with >1 million reactions (HPC)
- Development of new solvers in Julia, especially for large and multi-scale models
- Increased functionality of COBRA.jl
- **Collaborations welcome!**

References

1. Heirendt, L. et al. (2017) **DistributedFBA.jl: high-level, high-performance flux balance analysis in Julia**, Bioinformatics, 1-3, doi: 10.1093/bioinformatics/btw838.
2. Bezanson, J. et al. (2014) *Julia: A Fresh Approach to Numerical Computing*, arXiv:1411.1607 [cs.MS].
3. Duarte, N. C. et al. (2007) *Global reconstruction of the human metabolic network based on genomic and bibliomic data*, PNAS, 104(6), 1777-1782, doi: 10.1073/pnas.0610772104.
4. Ebrahim, A. et al. (2013) *COBRApy: COnstraints-Based Reconstruction and Analysis for Python*, BMC Systems Biology, 7(74).
5. Gudmundsson, S. et al. (2010) *Computationally efficient flux variability analysis*, BMC Bioinformatics, 11(1), 489.
6. Heinken, A. et al. (2015) *Systematic prediction of health-relevant human-microbial co-metabolism through a computational framework*, Gut Microbes, 6(2), 120-130.
7. Lubin, M. et al. (2015) *Computing in Operations Research using Julia*, INFORMS Journal on Computing, 27(2), 238--248, doi:10.1287/ijoc.2014.0623.
8. Magnusdottir et al. (2016) *Generation of genome-scale metabolic reconstructions for 773 members of the human gut microbiota*, Nature Biotechnology, advanced access, doi: 10.1038/nbt.3703.
9. Orth, J. D. et al. (2010) *Reconstruction and Use of Microbial Metabolic Networks: the Core Escherichia coli Metabolic Model as an Educational Guide*, EcoSal Plus, 1(10).
10. Palsson, B. et al. (2015) *Systems Biology: Constraint-based Reconstruction and Analysis*, Cambridge University Press, Edition 1.
11. Schellenberger, J. et al. (2011) *Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2.0,* Nature protocols, 6, 1290-1307.
12. Thiele, I. et al. (2013) *A community-driven global reconstruction of human metabolism*, Nature Biotechnology, 31, 419-425, doi:10.1038/nbt.2488.

github.com/opencobra/COBRA.jl





Acknowledgments

Sylvain Arreckx - Ines Thiele - Ronan Fleming

Systems Biochemistry & Molecular Systems Physiology Groups

Julia community