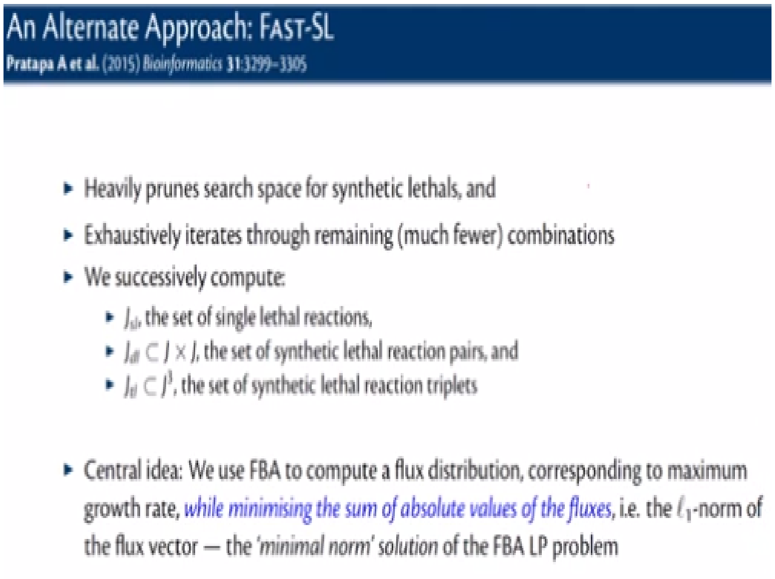
Computational Systems Biology
Karthik Raman
Department of Biotechnology
Indian Institute of Technology – Madras

Lecture - 69
Perturbations to Metabolic Networks: Synthetic Lethals

In today's video I will introduce you to how we compute synthetic lethals and metabolic networks based on the concepts of FBA, but you know using a very efficient algorithm developed in our lab, known as the Fast-SL algorithm. So, we came up with a new algorithm called Fast-SL in our lab and today we will discuss this algorithm.

The core of this algorithm is that we try to find a different FBA solution to start with and make some intelligent optimizations that helps us cut through the search space. So, instead of solving 170 million LPs, we end up solving a few 100 thousand LPs. That gives you a phenomenal savings in time and computation power.

(Refer Slide Time: 00:50)



An Alternate Approach: FAST-SL

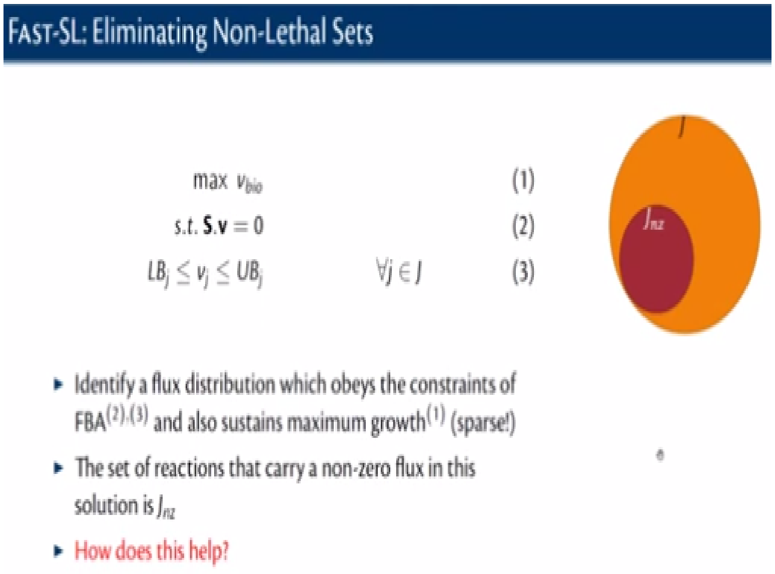
Pratapa A et al. (2015) *Bioinformatics* **31**:3299–3305

- Heavily prunes search space for synthetic lethals, and
- Exhaustively iterates through remaining (much fewer) combinations
- We successively compute:
  - J_{sl}, the set of single lethal reactions,
  - $J_{dl} \subset J \times J$, the set of synthetic lethal reaction pairs, and
  - $J_{tl} \subset J^3$, the set of synthetic lethal reaction triplets
- Central idea: We use FBA to compute a flux distribution, corresponding to maximum growth rate, *while minimising the sum of absolute values of the fluxes*, i.e. the ℓ_1-norm of the flux vector — the *'minimal norm' solution* of the FBA LP problem

Our approach involves heavily pruning the search space. How do we prune the search space, we will get to that? We then exhaustively iterate to the remaining combinations. So, it is in some sense related to the exhaustive enumeration, but involves only computing a very small fraction of the exhaustive set. So, we successively compute 3 sets, first we call Jsl, set of single lethals. If J is the set of all reactions, J x J is the set of all pairs of reactions and Jdl is the subset of that, Jtl is the subset of J3 in any 3 reactions from the total network.

The central idea is we use FBA to compute a particular flux distribution and we use this vector to further simplify the computations. We will get to what this vector is in a moment.

(Refer Slide Time: 01:52)



Fast-SL: Eliminating Non-Lethal Sets

$$\max\ v_{bio} \qquad (1)$$

$$\text{s.t. } \mathbf{S}.\mathbf{v} = 0 \qquad (2)$$

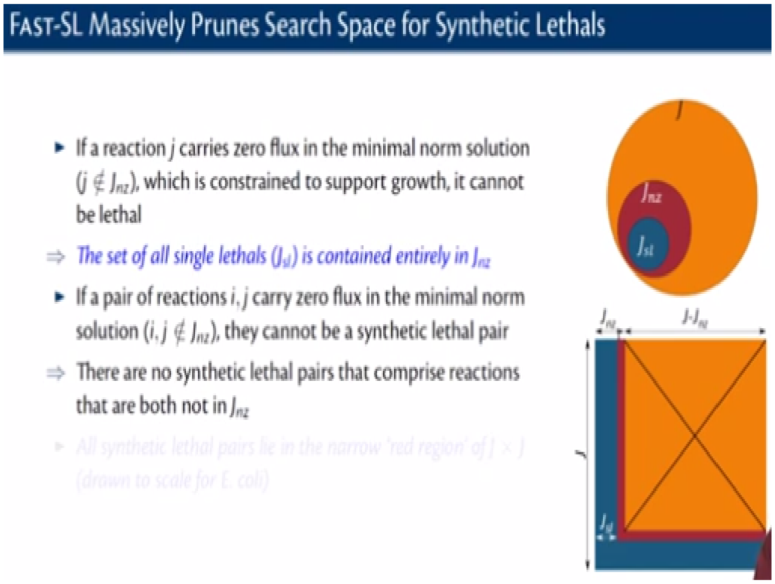
$$LB_j \leq v_j \leq UB_j \qquad \forall j \in J \qquad (3)$$

- Identify a flux distribution which obeys the constraints of FBA$^{(2),(3)}$ and also sustains maximum growth$^{(1)}$ (sparse!)
- The set of reactions that carry a non-zero flux in this solution is J_{nz}
- How does this help?

This is our good old linear program. What is FBA? Maximize V biomass such that SV=0, subject to some lower bounds for all reactions and upper bounds. We then identified our flux distribution, which obeys the constraints of FBA and also sustains maximum growth and let us say J that you see on the right hand side is the set of all reactions. The set of all reactions that carry a non-zero flux in a given FBA solution, let us call that Jnz for non-zero. This is a smaller subset of J.

In fact, this diagram that you see is drawn to scale for the E. coli model that we use. How does this help? We have some very interesting observations now, although it seems almost obvious in hindsight.

(Refer Slide Time: 02:45)



FAST-SL Massively Prunes Search Space for Synthetic Lethals

- If a reaction j carries zero flux in the minimal norm solution ($j \notin J_{nz}$), which is constrained to support growth, it cannot be lethal

⇒ *The set of all single lethals (J_{sl}) is contained entirely in J_{nz}*

- If a pair of reactions i, j carry zero flux in the minimal norm solution ($i, j \notin J_{nz}$), they cannot be a synthetic lethal pair

⇒ There are no synthetic lethal pairs that comprise reactions that are both not in J_{nz}

- *All synthetic lethal pairs lie in the narrow 'red region' of $J \times J$ (drawn to scale for E. coli)*

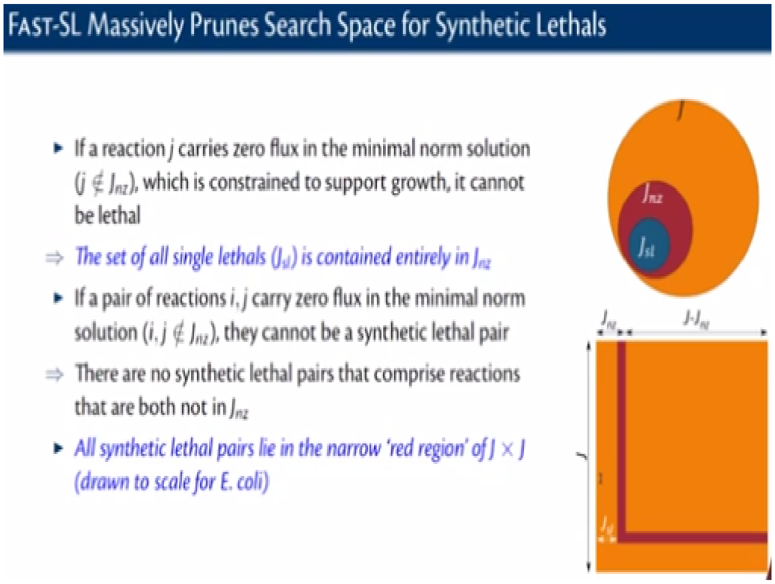
If a reaction J carries 0 flux in the FBA solution. If it carries 0 flux in the FBA solution, which is constrained to support growth, it cannot be lethal. This is a very simple sort of an obvious kind of thing. If a reaction is already carrying no flux, removing it is not going to kill the organism. That is very obviously reasonable. Which means there is no single lethal reaction outside of Jnz. That is the next jump.

This is the set of all reactions, all these reactions that are not in this dark portion, they are not carrying a flux. There are many possible solutions remember. There are many possible solutions to any FBA problem, but if it does not carry a flux in any solution it means that it cannot be a lethal reaction because the cell is growing without that reaction.

Therefore, we can say that all the single lethal reactions belong in Jnz. Because these guys are not carrying any flux, so all the single lethal reactions belong in Jnz. Which means that first of all you do not need to do J simulations like size J simulations, we just need to do size Jnz simulations to find synthetically single lethals in the first place. This means that the set of all single lethal reactions is contained entirely in Jnz fair enough.

The next step, let us consider all pairs of reactions, we can now represent in a matrix form, a J x J matrix and now we say that if a pair of reactions carry 0 flux in the minimal norm again they cannot be a synthetic lethal pair. That is a very obvious extension of the previous result. There are no reactions that are lethal in this orange part. Both reactions cannot be in Jnz. This is SL. These things will anyway not carry a flux. You remove a reaction which is singly lethal and you remove reaction from the cell, it is again not going to recover growth under this formulation. All these are anyway known lethals. So, let us leave that out too.

(Refer Slide Time: 05:18)



FAST-SL Massively Prunes Search Space for Synthetic Lethals

- If a reaction j carries zero flux in the minimal norm solution ($j \notin J_{nz}$), which is constrained to support growth, it cannot be lethal

⇒ *The set of all single lethals (J_{sl}) is contained entirely in J_{nz}*

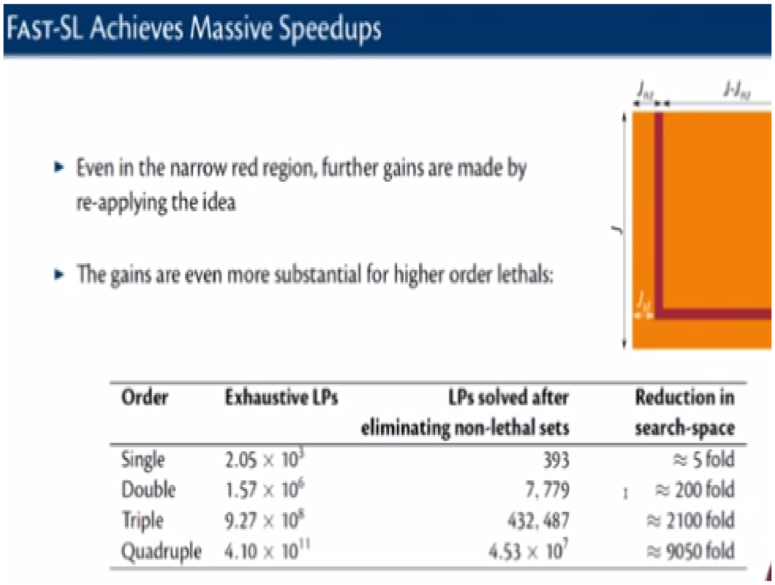
- If a pair of reactions i, j carry zero flux in the minimal norm solution ($i, j \notin J_{nz}$), they cannot be a synthetic lethal pair

⇒ There are no synthetic lethal pairs that comprise reactions that are both not in J_{nz}

- *All synthetic lethal pairs lie in the narrow 'red region' of $J \times J$ (drawn to scale for E. coli)*

All synthetic lethal pairs therefore now lie in this narrow region, which is again drawn to scale for E. coli.

(Refer Slide Time: 05:29)



FAST-SL Achieves Massive Speedups

- Even in the narrow red region, further gains are made by re-applying the idea
- The gains are even more substantial for higher order lethals:

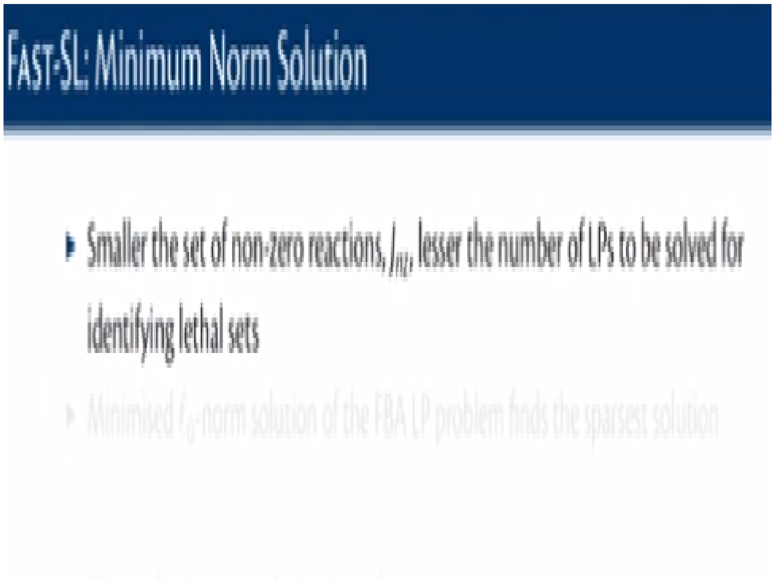
Order	Exhaustive LPs	LPs solved after eliminating non-lethal sets	Reduction in search-space
Single	2.05×10^3	393	≈ 5 fold
Double	1.57×10^6	7,779	≈ 200 fold
Triple	9.27×10^8	432,487	≈ 2100 fold
Quadruple	4.10×10^{11}	4.53×10^7	≈ 9050 fold

But if you carefully look at it, this is still not a major saving, there are more savings to be done by reapplying the idea and so on and the gains are even more substantial for higher order lethals.

So, let us just look at our results. We find that for single lethal, instead of solving about 2050 LPs, we solve only 393. That is already a five-fold saving, not a big deal. But, for double lethals, instead of solving 1.6 million LPs, we solve only about 8000 LPs, which is a 200-fold saving.

If you look at quadruple, the savings are astronomical. It is for something that would take weeks, what we done in hours, that is the level of simplification we have attained by making this search space slicing that we are doing.

(Refer Slide Time: 06:16)

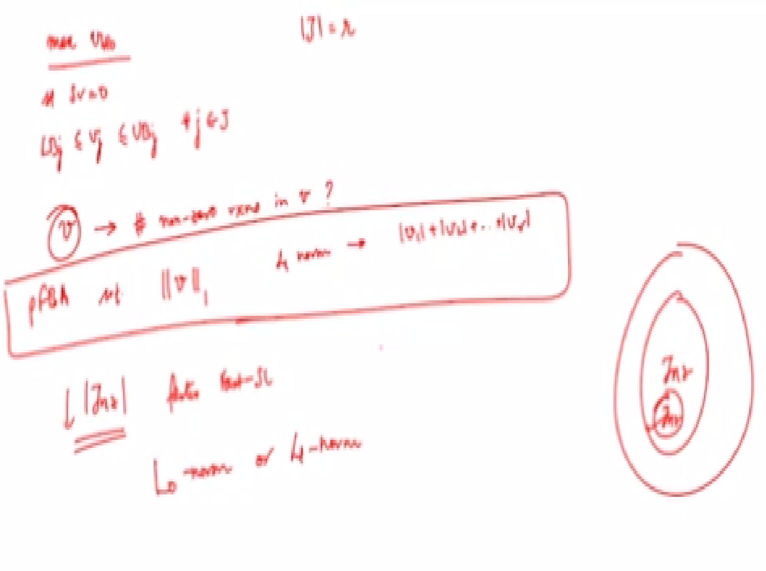


FAST-SL: Minimum Norm Solution

- Smaller the set of non-zero reactions, J_{nz}, lesser the number of LPs to be solved for identifying lethal sets
- Minimised l_0-norm solution of the FBA LP problem finds the sparsest solution

The smaller the set of non-zero reactions Jnz, the lesser the number of LPs to be solved. I did not tell you how we came up with this solution in the first place.

(Refer Slide Time: 06:35)


max v_{bio}
st $Sv=0$
$LB_j \le v_j \le UB_j \ \forall j \in J$
$|J| = r$
v → # non-zero rxns in v ?
pFBA st $\|v\|_1$
L_1 norm → $|v_1|+|v_2|+..+|v_r|$
$\downarrow |J_{nz}|$ faster Fast-SL
L_0-norm or L_1-norm
J_{nz}
J_{nz}

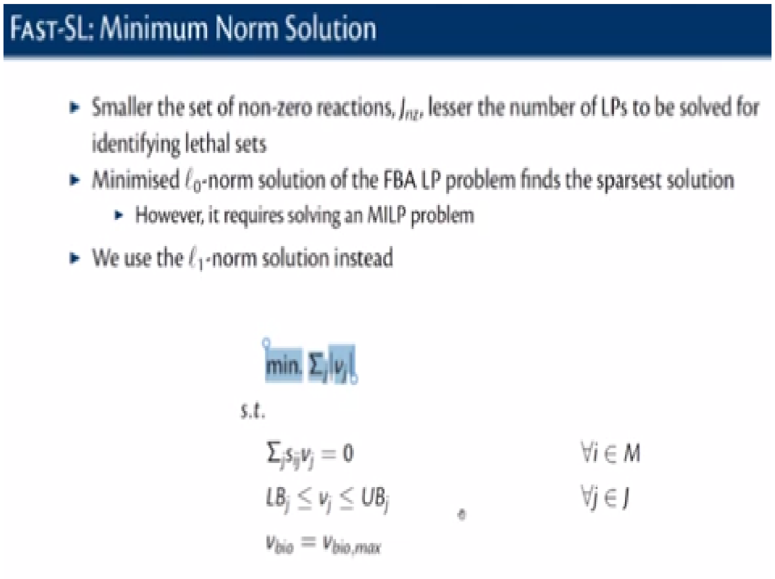
All I said was we have an FBA solution, so let us just write it out max Vbio such that SV=0 and some LBj <= Vj <= VBj for all J in the reaction set. So, the number of reactions is what we used to call R, the size of this reaction set. Now let us say you got a particular solution V. Now what is the number of non-zero reactions in V? This is something we need to find out.

Can you recall parsimonious FBA? In parsimonious FBA what did we do, we maximized Vbio such we would minimize the L1 or L0 norm of the vector. Normally, we want to ideally minimize the L0 norm that would be the real sparsest vector. But, because that also translates to an MILP problem, we prefer to do the L1 norm. Once again, what is L1 norm? This is the L1 norm, the absolute value of every element in the vector.

For the Fast-SL formulation, does it matter which one we pick. You can even take the regular FBA solution that you get without doing any of these. But, the smaller the size of Jnz, faster is Fast-SL. Because you get much more savings. What is this, you are trying to make this set smaller, so let us say if this was Jnz versus this as Jnz, this is going to give you much higher speeds, the smaller Jnz. How do you compute the smallest Jnz? You have to minimize the L0 norm or at least L1 norm, this is exactly what we do.

We stuck to the L1 norm for ease of computation and speed, but you can potentially use the L0 norm as well.

(Refer Slide Time: 09:40)



FAST-SL: Minimum Norm Solution

- Smaller the set of non-zero reactions, J_{nz}, lesser the number of LPs to be solved for identifying lethal sets
- Minimised ℓ_0-norm solution of the FBA LP problem finds the sparsest solution
  - However, it requires solving an MILP problem
- We use the ℓ_1-norm solution instead

$$\min. \textstyle\sum_j |v_j|$$

s.t.

$$\textstyle\sum_j s_{ij} v_j = 0 \qquad \forall i \in M$$

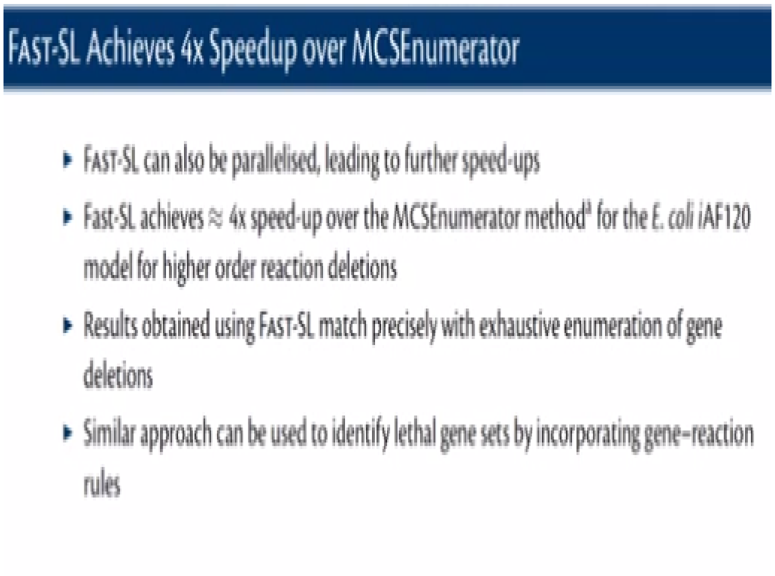
$$LB_j \leq v_j \leq UB_j \qquad \forall j \in J$$

$$v_{bio} = v_{bio,max}$$

The minimized L0 norm solution of the FBA LP problem finds the sparsest solution; however, that requires solving an MILP. We instead use the L1 norm, which translates to this. So, minimize the sum of all fluxes in your final biomass flux such that your FBA conditions are all satisfied, your mass balance, your bounds and so on and additionally Vbio is the same as what you obtained in the first FBA, where you tried to maximize.

Amongst the alternate candidate solutions of the FBA problem, which all give the same function value, Vbio max, can you find the one which has the smallest sum of fluxes, ideally smallest number of fluxes, but that being a costly problem we solve for smallest sum of fluxes.

(Refer Slide Time: 10:33)



FAST-SL Achieves 4x Speedup over MCSEnumerator

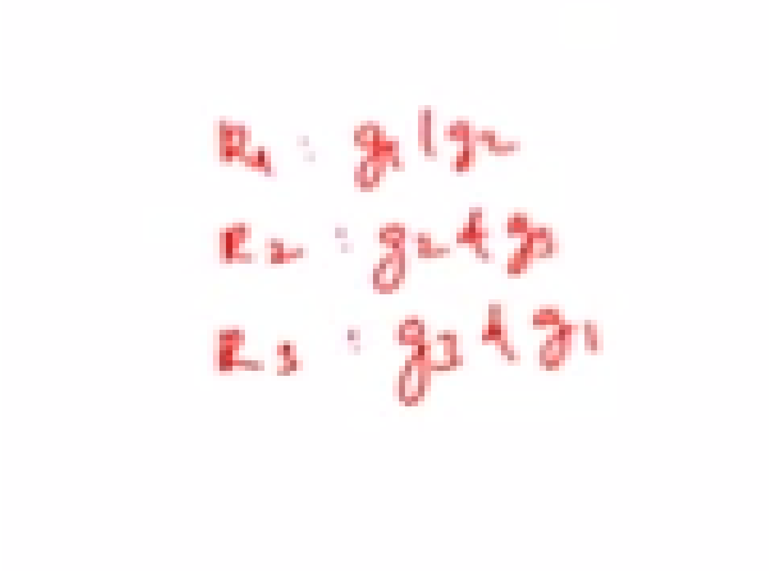
- FAST-SL can also be parallelised, leading to further speed-ups
- Fast-SL achieves ≈ 4x speed-up over the MCSEnumerator method[a] for the *E. coli* iAF120 model for higher order reaction deletions
- Results obtained using FAST-SL match precisely with exhaustive enumeration of gene deletions
- Similar approach can be used to identify lethal gene sets by incorporating gene–reaction rules

It is possible, that is a good point, in practice L1 norm works well for us. I would say that ideally we need to go for L0 norm, but we need to do this multiple times during the course of a double lethal analysis. So, you do not want to end up solving so many L0 based problems. Unless it is a pathological model where the savings are very different between L0 and L1, I do not think this is an issue.

For all the organisms that we tried L0 and L1 were not a big deal. There was obvious a difference may be 6, 7, 8, 10, but single or low double digits. Really no big major difference. So, Fast-SL finally achieved four times speed up our MCS enumerator. Much higher speed up our earlier reported SL finder, but MCS enumerator is the more recent algorithm and we also tested this with exhausted enumeration and it turned out that the results obviously agreed exactly. We have proof for the algorithm as well.

A very similar approach can be used to find synthetic lethal gene sets as well. The only thing with gene sets is it is a little more trickier.

(Refer Slide Time: 11:44)



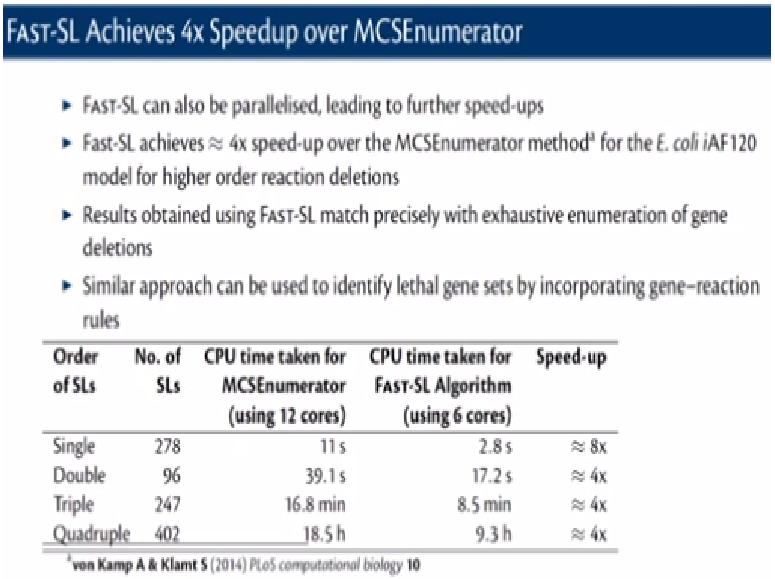
$R_1 : g_1 | g_2$

$R_2 : g_2 \& g_3$

$R_3 : g_3 \& g_1$

Remember that you could have R1 is g1 or g2, R2 is g2 and g3. So, when you remove g1, nothing gets affected, but when you remove g3, R2 gets affected and may be even R3 gets affected. You have to worry about what reactions go out when you remove a pair of genes and we did discuss this sometime back you can refresh your memory. But this involves incorporating all the gene protein reaction rules.

(Refer Slide Time: 12:17)



FAST-SL Achieves 4x Speedup over MCSEnumerator

- FAST-SL can also be parallelised, leading to further speed-ups
- Fast-SL achieves ≈ 4x speed-up over the MCSEnumerator method[a] for the *E. coli* iAF120 model for higher order reaction deletions
- Results obtained using FAST-SL match precisely with exhaustive enumeration of gene deletions
- Similar approach can be used to identify lethal gene sets by incorporating gene–reaction rules

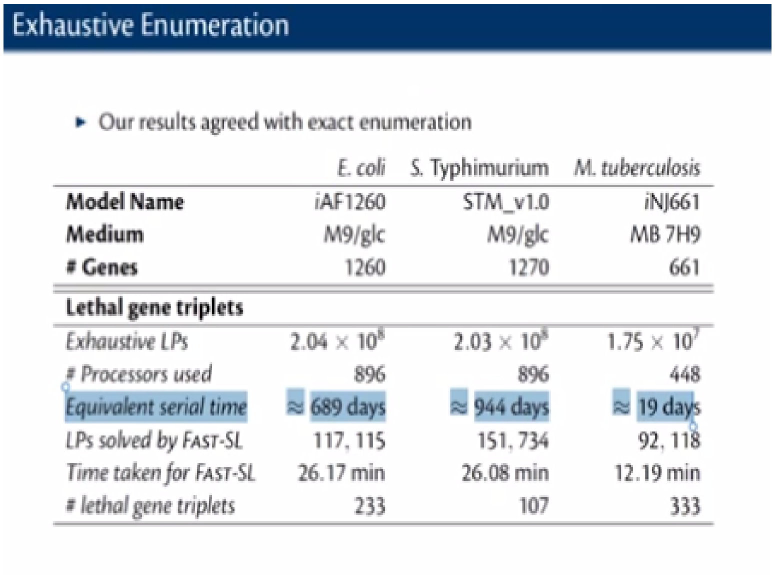
Order of SLs	No. of SLs	CPU time taken for MCSEnumerator (using 12 cores)	CPU time taken for FAST-SL Algorithm (using 6 cores)	Speed-up
Single	278	11 s	2.8 s	≈ 8x
Double	96	39.1 s	17.2 s	≈ 4x
Triple	247	16.8 min	8.5 min	≈ 4x
Quadruple	402	18.5 h	9.3 h	≈ 4x

[a] **von Kamp A & Klamt S** (2014) *PLoS computational biology* **10**

This was our final result table, so for E. coli we got about 4x speed up on average for double, triple and quadruple gene deletions. It took us only about 8.5 minutes to identify triple gene deletions. If you recall the SL finder it was taking about a week, but of course MCS enumerator takes only about 17-18 minutes, so the speed up is almost 4x, but the thing is the scale is very well.

So we are not sure how MCS enumerator scales and it depends upon how scalable the MILP algorithm itself is, whereas in our case we does have many simple problems to be solved, way about a few 100 thousand LPs to be solved.

(Refer Slide Time: 13:03)



Exhaustive Enumeration

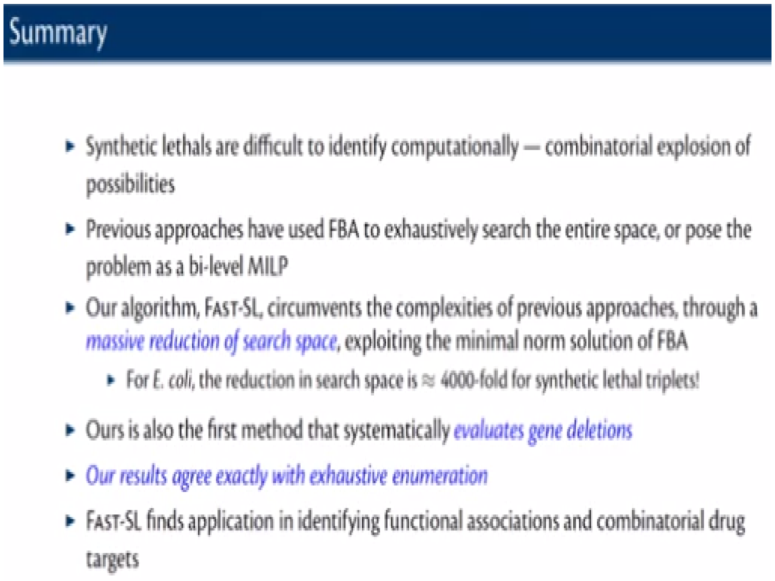
- Our results agreed with exact enumeration

	E. coli	*S. Typhimurium*	*M. tuberculosis*
Model Name	iAF1260	STM_v1.0	iNJ661
Medium	M9/glc	M9/glc	MB 7H9
# Genes	1260	1270	661
Lethal gene triplets			
Exhaustive LPs	2.04×10^8	2.03×10^8	1.75×10^7
# Processors used	896	896	448
Equivalent serial time	≈ 689 days	≈ 944 days	≈ 19 days
LPs solved by FAST-SL	117, 115	151, 734	92, 118
Time taken for FAST-SL	26.17 min	26.08 min	12.19 min
# lethal gene triplets	233	107	333

This was the approximate serial time that we took. It took about 19 days for tuberculosis, whereas we needed only about 12 minutes. This took about like 900 days and we needed about 26 minutes and so on. Of course this is serial, so it is not an interesting comparison, but just to let you know that we have done that and basically double verified our claims. This was across 896 so this finally took a day for us.

If you were to put so many processes for Fast-SL, it will terminate in half a second or something like that. Because we are still solving a lot of parallel LPs. In fact, the way the algorithm works is that we assemble a list of LPs to solve and then distribute it and solve. The first assembling involves doing some preliminary LPs, Jnz computation and things like that.

(Refer Slide Time: 13:58)



Summary

- Synthetic lethals are difficult to identify computationally — combinatorial explosion of possibilities
- Previous approaches have used FBA to exhaustively search the entire space, or pose the problem as a bi-level MILP
- Our algorithm, Fast-SL, circumvents the complexities of previous approaches, through a *massive reduction of search space*, exploiting the minimal norm solution of FBA
  - For *E. coli*, the reduction in search space is ≈ 4000-fold for synthetic lethal triplets!
- Ours is also the first method that systematically *evaluates gene deletions*
- *Our results agree exactly with exhaustive enumeration*
- Fast-SL finds application in identifying functional associations and combinatorial drug targets

Synthetic lethals I hope I have convinced you that they are difficult to identify computationally because of the combinatorial explosion of possibilities. The previous approaches I have used FBA to exhaustively search the entire space or pose it as a bi-level MILP. We avoid both the MILP formulation and exhaustive enumeration, but our approach inspired by exhaustive enumeration, where in we cut down the search space and finally have about a 4000-fold improvement for synthetic lethal triplets.

We also systematically evaluate gene deletion. The previous studies had only looked at reaction deletions. This involves some intelligent handling of the GPR constraints and things like that and our results agree exactly with exhaustive enumeration. The applications are in finding functional associations and combinatorial drug targets. You might be able to find some really obscure functional associations.

Three genes in different pathways could actually be involved together because they involve the production of the same metabolite or something like that in some way. The other thing I remember is that synthetic lethals are environment specific. So, all these were done in a minimal glucose medium, but we can potentially compute synthetic lethals in any different environment.

In today's video, I hope you got a good overview of the Fast-SL algorithm, which is a very different take on synthetic lethals in metabolic networks. It tries to slice through the search space by exploiting the alternate optima of FBA essentially and we get a very good speed up almost we end up solving only 1 in 4000 simulations when you are looking at triple gene deletions and so on. So it is a very efficient method which can also be easily parallelized to compute synthetic lethals.

In the next video, we will start winding up with constraint based approaches. We will first look at the limitations, essentially what are all the things that can go wrong when you predict lethality or growth using constraint based approaches.