mergem: merging and comparing genome-scale metabolic models using universal identifiers

Archana Hari and Daniel Lobo[*]

Department of Biological Sciences
University of Maryland, Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250, USA.

[*] Corresponding author
E-mail: lobo@umbc.edu
Tel: (410) 455-5726

Abstract

Numerous methods exist to produce and refine draft genome-scale metabolic models. However, there is a lack of automated tools that can integrate these drafts into a curated single comprehensive model. In addition, computing and visualizing metabolic differences and similarities, including reconstructions using incompatible identifiers, is a current challenge. Here we present *mergem*, a novel method to compare and merge two or more metabolic models using a universal metabolic identifier mapping system constructed from multiple metabolic databases. *mergem* is implemented as a Python package and on the web-application Fluxer, which allows simulating and comparing multiple models with different interactive flux graphs.

Keywords:

Genome-scale metabolic models, model reconstruction, pathway comparison, network visualization, database integration

1. Background

Genome scale metabolic models (GEMs) are *in silico* descriptions that can represent and simulate the metabolic networks of biological systems at the cellular or even organismal level. Each model comprises a set of mathematically formulated gene-protein-reaction relationships that contribute to the metabolic state of the biological system [1,2]. There are four major stages for reconstructing GEMs: draft reconstruction, conversion to mathematical format, refinement, and network evaluation. The last three steps are typically iterated until predictions match experimental findings and the validated model is considered a robust GEM [3]. Robust GEMs should be constructed using all the information available at the time of reconstruction, with the goal to be metabolically complete [4]. Towards this, several pipelines and approaches to generate draft reconstructions are available, including fully automatic tools [5]. However, reconstructions built from the same genome using different tools frequently result in different metabolic coverage due to differences in the underlying algorithms and reaction database they use [6]. Merging such draft reconstructions for the same organism into a single comprehensive model can increase the metabolic coverage, leverage the advantages of different reconstruction tools, and thereby improve the completeness of the resulting model. Yet, the large size and connectivity of GEMs and the use of different identifier namespaces—for reactions, metabolites, and genes—by different pipelines makes the merging and visualization of different reconstructed GEMs a current challenge [7,8].

A number of tools are available that can merge GEMs and metabolic reconstructions, but their functionality is limited. MetaMerge [9] is a Python library that can take two metabolic networks as input and unify their metabolites and reactions based on their features such as metabolite CAS number, KEGG identifier, or IUPAC name and reaction name, its gene name, or pathway name. modelBorgifier [10] is a MATLAB toolbox for matching and comparing, semi-automatically, two given genome-scale reconstructions by merging metabolites and reactions based on a set of parameters such as metabolite name, KEGG ID, reaction name and the number of reactants and products in reaction. iMet [11] is a graphical user interface software that merges metabolic networks in a semi-automatic fashion. The MetaNetX web-interface [12] provides tools for combining models and identifying common and unique components between pairs of models. The COBRApy [13] package also offers a merge function, but it is based on merging reactions and metabolites with the same identifiers. Although these tools are very useful for processing GEMs, they have important limitations that reduce their applicability (see section 2.4 for a detailed comparison). In brief, all these tools can merge only two models at a time, have problems merging models with different database identifiers, can take a long time to merge two models, do not offer a visual comparison, and most of them require the user to be familiar with a programming language such as MATLAB or Python.

The comparison and merging of models can be aided by visualizing the commonalities and differences of metabolite and reaction components in each model. Such graphical representations can be beneficial not only for unifying draft reconstructions from different pipelines, but also to discover differences between cells and organisms with validated GEMs. While tools such as Escher [14], MetExplore [15], and Pathview [16] can be used to visualize GEMs, they are not amenable for comparing the differences between multiple models or merging them. There is thus a need for a new approach that can robustly compare, merge, and visualize multiple reconstructions produced from different pipelines with different database identifiers and without keeping duplicates or causing loss of information.

Here we present a novel methodology and software tool called *mergem* for comparing and merging GEMs and draft reconstructions. The proposed algorithm translates and unifies metabolite identifiers using a comprehensive database mapping system and then compares reactions based on their participating reactant and product metabolites. The method is freely available as a Python package and command line tool. In addition, *mergem* has been integrated into the user-friendly Fluxer web application [17], which allows any user without programming knowledge to perform the merging, comparison, and visualization of any number of models directly in the web application. Figure 1 illustrates the main steps of the proposed methodology. We demonstrate the application of the method with several use-cases, including merging multiple draft reconstructions and comparing published GEMs from different organisms. The proposed merging algorithm and tool and its integration with Fluxer's robust metabolic network visualization and analysis capabilities can not only streamline the reconstruction of robust GEMs but also provides a user-friendly interface to visually compare flux networks of different reconstructions and organisms, as well as facilitates the generation of hypotheses for metabolic engineering, biomedicine, and drug development applications.

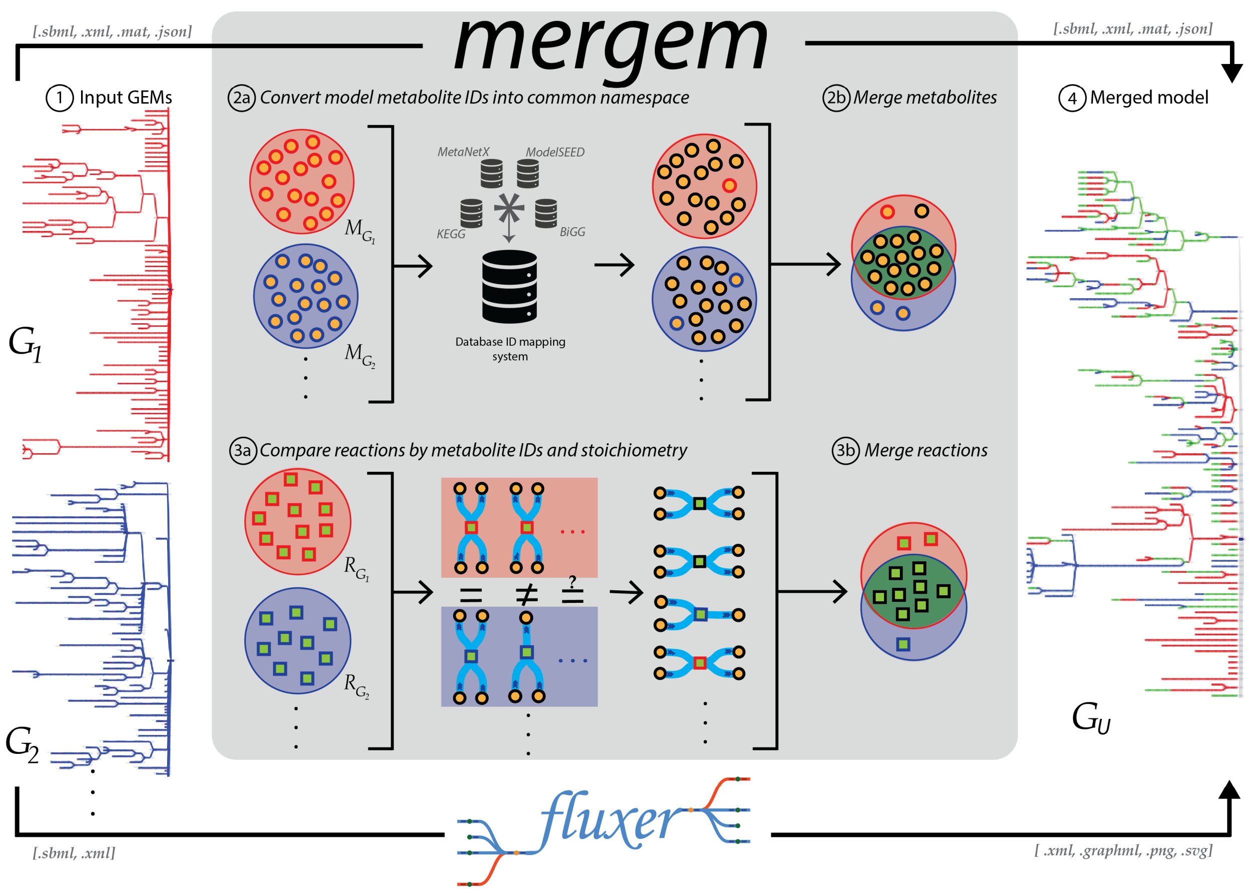

mergem
[.sbml, .xml, .mat, .json]
[.sbml, .xml, .mat, .json]
1 Input GEMs
2a Convert model metabolite IDs into common namespace
2b Merge metabolites
4 Merged model
MetaNetX
ModelSEED
KEGG
BiGG
Database ID mapping system
M_{G_1}
M_{G_2}
G_1
G_2
3a Compare reactions by metabolite IDs and stoichiometry
3b Merge reactions
R_{G_1}
R_{G_2}
G_U
fluxer
[.sbml, .xml]
[.xml, .graphml, .png, .svg]

Figure 1. *Key steps of the mergem method for merging genome-scale metabolic models (GEMs), as implemented in a command-line tool, a Python package, and Fluxer user-friendly web application.* ***1.*** *Two or more GEMs are taken as input.* ***2a.*** *The model metabolite identifiers (IDs) are mapped to a universal common namespace.* ***2b.*** *Metabolites with the same universal identifier are merged.* ***3a.*** *Reactions are compared using reactant and product metabolite universal identifiers.* ***3b.*** *Similar reactions are merged.* ***4.*** *The resultant merged model is returned, along with merging information and a Jaccard matrix measuring the distance between each pair of input models.*

2. Results

2.1. Universal mapping of metabolite and reaction identifiers

Different draft reconstruction pipelines label metabolites and reactions with different sets of identifiers, which preclude their direct use for merging metabolites and reactions. Various efforts including ModelSEED biochemistry [18], MetaNetX/MNXref [19,20] and BiGG [21] exist to overcome the identifier inconsistency problem by including mappings from their native identifiers with those from other resources. However, there is no streamlined method to apply such mappings from different databases, which also contain inconsistencies [7,8].

To overcome the problems with multiple identifier systems and inconsistent mappings, we developed an algorithm and implemented it in *mergem* to automatically process and reconcile database identifiers, resulting in a universal mapper for metabolites and reactions. The tool downloads the most recent data from the main databases for metabolic information, including MetaNetX, ModelSEED, BiGG, KEGG [22], and ChEBI [23], and iteratively processes their metabolite and reaction information, as well as cross-reference mappings. Metabolite and reaction entities are extracted from each database and assigned a unique identifier (called universal ID) linked to their listed set of properties (name, molecular weight, InChI key [24], chemical formula, and E.C. number). Table 1 shows the specific metabolite and reaction properties obtained by *mergem* from each database.

Next, the algorithm processes the cross-reference mappings listed in each database as pairs of metabolite or reaction identifiers linking one native database identifier (source) with an identifier from another database (target). Table 1 shows the source and target databases used for identifier mappings (secondary identifiers and mappings from ModelSEED are ignored due to the many inconsistencies found). When the algorithm finds a cross-reference pair that corresponds to different universal IDs, the two universal identifiers and their properties are merged into a single entity. To avoid mapping conflicts when two different identifiers of the same database are cross-referenced to the same universal ID, priority is given to the first database where the cross-reference was found—hence, the order of the input databases establishes precedence. The algorithm finally outputs the built metabolite and reaction mapper linking the identifiers found in any of the databases processed to their universal ID and combined properties. Metabolite and reaction identifiers mapping to the same universal ID are considered synonymous. In this way, the universal IDs are used as an internal common namespace to efficiently translate metabolite identifiers from different databases and hence enable merging metabolite and reactions across GEMs from different pipelines.

Table 1*. mergem automatically downloads, processes, and consolidates metabolite (blue dot) and reaction (red dot) properties and identifiers from source databases into a universal identifier mapper for merging models.*

		Source Database				
		KEGG	MetaNetX	BiGG	ModelSEED	ChEBI
Property	Name	🔵	🔵	🔵🔴	🔵🔴	🔵
	Formula	🔵	🔵		🔵	
	Mass	🔵	🔵		🔵	
	E.C. Number		🔴	🔴	🔴	
	InChIkey		🔵	🔵	🔵	🔵
	Cross-ref Link			🔵🔴		
Cross-reference Target Database	BiGG		🔵🔴	🔵🔴	🔵	
	Biocyc			🔵		
	ChEBI	🔵	🔵	🔵		
	HMDB		🔵	🔵		
	KEGG	🔵	🔵🔴	🔵	🔵	
	LIPID MAPS		🔵	🔵		
	MetaNetX		🔵🔴	🔵🔴	🔵	
	Metacyc		🔵🔴			
	ModelSEED		🔵🔴	🔵🔴	🔵🔴	
	Reactome		🔵	🔵		
	Rhea		🔴			
	SLM		🔵			
	SabioRK		🔵🔴			

Source databases are updated regularly, and the most recent run of the mapping algorithm processed a total of 2,532,674 metabolite and 299,075 reaction identifiers, unifying them into 1,306,757 metabolite and 75,125 reaction unique universal identifiers. Figure 2 shows the resultant number of cross-referenced identifiers that were mapped by the algorithm. Each of these identifiers can thus be recognized by *mergem* and is associated with a unified set of metabolite or reaction properties, including name, InChI key, molecular weight, E.C. number, and chemical formula. The universal mappings and properties can be retrieved and used by third-party applications through the *mergem* Python library.

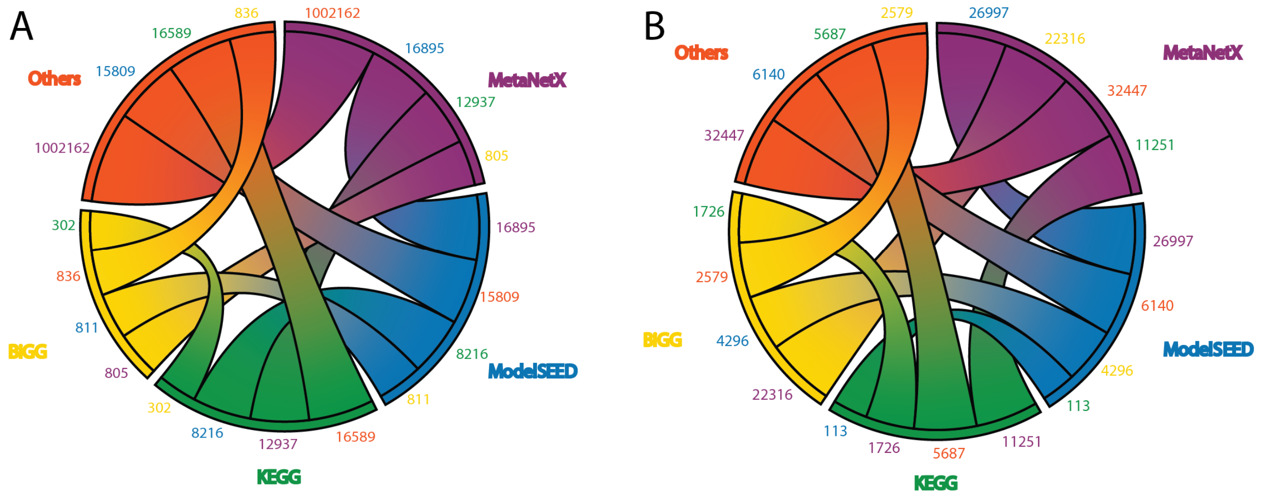


Figure 2. *Number of synonymous metabolites (A) and reactions (B) identifiers cross-referenced between databases by the universal mapping system in mergem.*

2.2. Method for merging GEMs

Based on *mergem* universal identifier mapper (see previous section), we developed an algorithm to merge the metabolites and reactions of two or more GEMs using the same or different identifier namespaces. The algorithm is formally defined in the methods section. In brief, the algorithm uses the first model as a template to which the metabolites and reactions from other models are merged into. Metabolite identifiers are mapped to a common namespace (*mergem* ID) as the concatenation of their universal identifier and cellular localization. Reaction keys are generated based solely on the set of reactants and products to guide the matching of equivalent reactions among the models. Multiple metabolite IDs can map to the same *mergem* ID, for which the algorithm maximizes metabolite and reaction matches while guaranteeing that all metabolites and reactions from the first model—the template—are present intact in the final merged model (see methods section for details and pseudocode). The objective function for the merged model can be set either by copying the objective function from one the input models or by merging the objective functions from all the input models.

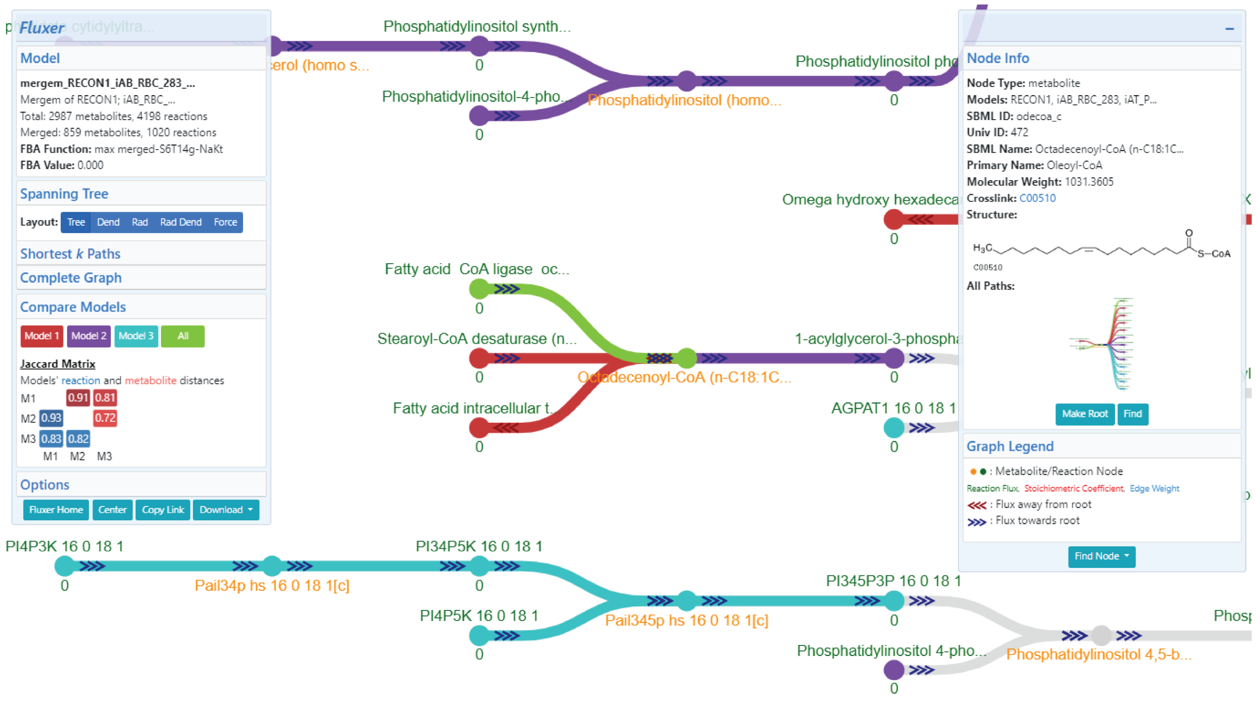
Once all the input models have been merged, the metabolite identifiers are reverted to the original identifier from the first model containing it, hence preserving the original identifier namespace. The algorithm returns the merged model along with source dictionaries mapping the reactions and metabolites from the input models to the merged model, which can be used for analyzing the commonalities and differences between models. Additionally, a matrix of Jaccard distances is computed and returned by the algorithm (see methods section) that quantifies the similarity between each pair of input models in terms of metabolites (matrix elements above the diagonal) and reactions (matrix elements below the diagonal). The range of each Jaccard distance element is [0,1], where 0 indicates all metabolites or reactions are similar between the models and 1 indicates no common metabolite or reaction exists between the models.

2.3. *mergem* package and web application

The presented method was implemented as an open-source Python package called *mergem*. It can be imported into Python scripts or executed as a command-line tool. Models can be input and output in a variety of formats, including SBML [25], MATLAB (The Mathworks, Inc.), and JSON. The efficient implementation of the algorithm can merge standard GEMs in seconds and multiple large models in minutes using a regular desktop computer.

In addition to the Python package, *mergem* has been incorporated into the freely-available web application Fluxer [17], which can create, visualize, and simulate flux networks from GEMs. Fluxer simulates input models with Flux Balance Analysis (FBA) [26] and uses the resulting flux values to compute flux networks that can be visualized as spanning trees, *k*-shortest paths, and complete graphs with multiple layouts. The graphs are interactive and the user-friendly interface also allows users to explore large metabolic networks and perform simulations of reaction knock-outs. With the integration of *mergem,* the web-application now allows any user to visually merge and compare any number of GEMs from those already curated in the application—including the models from the BiGG database [27] or previously created in Fluxer through their private URL—or directly from SBML files that can be easily uploaded to the application. Similar to the *mergem* Python package, users can input any number of models for merging, and either select an objective function from any of the input models or merge all objective functions into a single objective reaction.

Figure 3 displays the Fluxer interface after merging three human models: RECON1 [28], red blood cell (iAB_RBC_283 [29]) and platelets (iAT_PLT_636 [30]). The information card on the left includes merging statistics, such as the number of metabolites and reactions merged between any of the input models and the Jaccard distance matrix. The elements below the matrix diagonal (blue) represent the Jaccard distance between the reactions for each pair of models, while the elements above the matrix diagonal (red) represent the Jaccard distance between the metabolites for each pair of models. The diagonal elements are always zero by definition and hence not shown. The intensity of the element colors in the matrix indicates how similar (lighter) or different (darker) the reaction or metabolite sets are between the pair of models. The Fluxer graphs display metabolites, reactions, and their connections in different colors depending on whether they are unique to a particular model (red, purple, and blue in the figure), common to all models (green), or contained in two or more models but not all (gray). Colors can be customized by the user. Clicking on a metabolite or reaction displays a card on the right showing its properties, such as names, molecular weight, links to other databases, and molecular structure, as well as which input models contain that particular reaction or metabolite. Fluxer can simulate the knock-out of reactions in the merged model, after which a new FBA analysis is performed and a new flux graph is computed and displayed. Flux graphs of the merged model can be downloaded as images, vector graphics, and GraphML, and the merged model can be downloaded as SBML. The interface also includes the ability to perform and download a FROG analysis [31], which includes flux variability analysis, reaction deletion fluxes, objective function values, and gene deletion fluxes. Importantly, the URL generated for a merged model is unique and persistent, which can be used to share publicly or privately the merged model including all the interactive functionality and analysis tools in Fluxer.


Fluxer

Model

mergem_RECON1_iAB_RBC_283_...
Mergem of RECON1; iAB_RBC_...
Total: 2987 metabolites, 4198 reactions
Merged: 859 metabolites, 1020 reactions
FBA Function: max merged-S6T14g-NaKt
FBA Value: 0.000

Spanning Tree

Layout: Tree Dend Rad Rad Dend Force

Shortest *k* Paths

Complete Graph

Compare Models

Model 1 Model 2 Model 3 All

Jaccard Matrix
Models' reaction and metabolite distances

	M1	M2	M3
M1		0.91	0.81
M2	0.93		0.72
M3	0.83	0.82	

Options

Fluxer Home Center Copy Link Download

Node Info

Node Type: metabolite
Models: RECON1, iAB_RBC_283, iAT_P...
SBML ID: odecoa_c
Univ ID: 472
SBML Name: Octadecenoyl-CoA (n-C18:1C...
Primary Name: Oleoyl-CoA
Molecular Weight: 1031.3605
Crosslink: C00510
Structure:

All Paths:

Make Root Find

Graph Legend

●● : Metabolite/Reaction Node
Reaction Flux, Stoichiometric Coefficient, Edge Weight
<<< : Flux away from root
>>> : Flux towards root

Find Node

cytidylyltra...
Phosphatidylinositol synth...
erol (homo s...
Phosphatidylinositol-4-pho...
Phosphatidylinositol (homo...
Phosphatidylinositol pho
Omega hydroxy hexadeca
Fatty acid CoA ligase oc...
Stearoyl-CoA desaturase (n...
Octadecenoyl-CoA (n-C18:1C...
1-acylglycerol-3-phospha
Fatty acid intracellular t...
AGPAT1 16 0 18 1
PI4P3K 16 0 18 1
Pail34p hs 16 0 18 1[c]
PI34P5K 16 0 18 1
PI4P5K 16 0 18 1
Pail345p hs 16 0 18 1[c]
PI345P3P 16 0 18 1
Phosphatidylinositol 4-pho...
Phosphatidylinositol 4,5-b...
Phos
0
H_3C
S–CoA
O
C00510

Figure 3*. Fluxer interactive web interface after merging and comparing three eukaryotic models: global human metabolic network RECON1, erythrocyte metabolism iAB_RBC_283, and platelet metabolism iAT_PLT_636. Nodes and edges in red, purple, and cyan are unique to RECON1, red blood cells, and platelets, respectively; elements in green are common to all models; and elements in gray belong to only two models. The left card shows details of the merged models and the Jaccard matrix summarizing the similarity between each model pair. Buttons to modify and customize the computed graph are also in the left card. The right card displays properties about the selected metabolite or reaction (Oleoyl-CoA in the figure) such as the name, molecular mass, and the universal identifier. Buttons to root, find, or knock-out the selected node are also available in the right card.*

2.4. Features and performance compared to other tools

Existent tools for merging or comparing GEMs differ in their methodology, their type of interface, the number of file inputs and formats, and their ability to visually display the results. Table 2 compares the main features of the currently-available automatic merging tools for GEMs. The main differences are that most tools can merge only pairs of models, are not able to merge models that utilize different identifier systems for metabolites and reactions, and cannot visually present or interact with the results. MetaMerge [9] was excluded from this study since it is scripted in an outdated version of Python and the package is missing files. ModelBorgifier [10] was also omitted since the method requires the user to manually reconcile metabolites and reactions.

Table 2*. Key features of currently-available automatic tools for merging and comparing GEMs.*

Tool	**Merging method**		**Type**	**Max. no. models**	**Model input format**	**Visualization**
	Metabolites	**Reactions**				
COBRApy [26]	Direct ID string matching	Direct ID string matching	Python library	2	COBRApy model objects	-
MetaNetX [19]	Internal ID mapper	Identifier based	Web app	2	SBML files following specific standard	-
iMET [11]	Identifier based – connects to KEGG for more information before merging	Identifier based – connects to KEGG for more information before merging	GUI native app	2	SBML files	-
mergem (This study)	Universal ID mapper	Universal metabolite IDs and their presence as product or reactant	Python command line	Any	SBML, MATLAB, JSON files	-
			Python library	Any	SBML, MATLAB, JSON files, and COBRApy model objects	-
Fluxer (This study)	*mergem* algorithm	*mergem* algorithm	Web app	Any	SBML files, select from curated models, URLs to fluxer models	Interactive color-coded visualization

COBRApy [26] is a Python implementation of COBRA and includes a merging tool that can take as input two COBRA model objects and output a combined model. The merging function performs a direct string matching of the metabolites and reactions identifiers to compare pairs of models, which can lead to disconnected networks, duplicated metabolites and reactions, or erroneous deletion of elements in the resulting model. The *mergem* algorithm instead uses a universal mapping dictionary that can translate different database identifiers for metabolites to a common namespace. *mergem* does not use reaction identifiers and instead compares the participating reactant and product metabolites to find matching reactions, bypassing completely the identifier mapping problem for reactions.

iMET is a standalone graphical user interface that can semi-automatically merge metabolic networks in the SBML file format [11]. The algorithm uses features such as metabolite name and KEGG ID to compare pairs of metabolites after which it assigns a similarity score to each pair. Reactions pairs are also assigned a similarity score by comparing their features, such as reactants and products. The similarity scores are used to reconcile the metabolites and reactions between the two models and users can choose to keep the result for each entity or manually change the merging. This method however can only merge two models at a time, requires the two models to follow the same SBML version, and heavily depends on the information provided within the SBML file itself or information extracted from KEGG (if users select this option) and thus can take hours to finish merging a pair of models. *mergem* does not have those restrictions or dependencies and takes a few minutes to merge very large models.

MetaNetX is a website with a repository of GEMs that also provides tools to construct, compare, analyze, and simulate GEMs [12]. There is no single tool on MetaNetX that helps identify both the common and unique reactions and metabolites between models. Importing each model individually and performing such comparative analyses involves multiple steps on MetaNetX. Further, only two models can be merged at a time on MetaNetX. *mergem* however can perform simultaneously in a single step the merging and statistical analysis for any number of models.

To evaluate their performance, we used each tool for pairwise-merging different draft models of *Pseudomonas putida* that were constructed with six different pipelines: AuReMe [32], Pathway Tools [33], CarveMe [34], RAVEN [35], ModelSEED [18], and MetaDraft [36] as reported by Mendoza et al. [37]. Figure 4 summarizes the metabolite and reaction merging results for each tool and pair of models (see Supplementary Table 1 for values). Each color represents a different tool, while empty circles indicate that the tool failed to complete the merging or load one of the input models. These results show that *mergem* is not only able to merge models from various reconstruction pipelines using different identifier naming systems but also merge most metabolic components. In contrast, iMET and MetaNetX were able to merge only four and three model pairs, respectively. Only the AuReMe and Pathway Tools reconstructions could be merged with all four tools. Visualizing the resulting merged models revealed disconnection in the metabolic networks merged using COBRApy and iMET. Further, the iMET merged models use their own identifiers and do not follow any database standard. *mergem* thus outperformed the other three tools based on the metabolites and reactions merged, identifiers in the merged model, and the extent of integration of the two input models. These results illustrate the importance of using a method more complex than directly matching model identifiers and how *mergem* can overcome this problem by using a universal mapper and metabolite-focused approach to match metabolites and reactions.

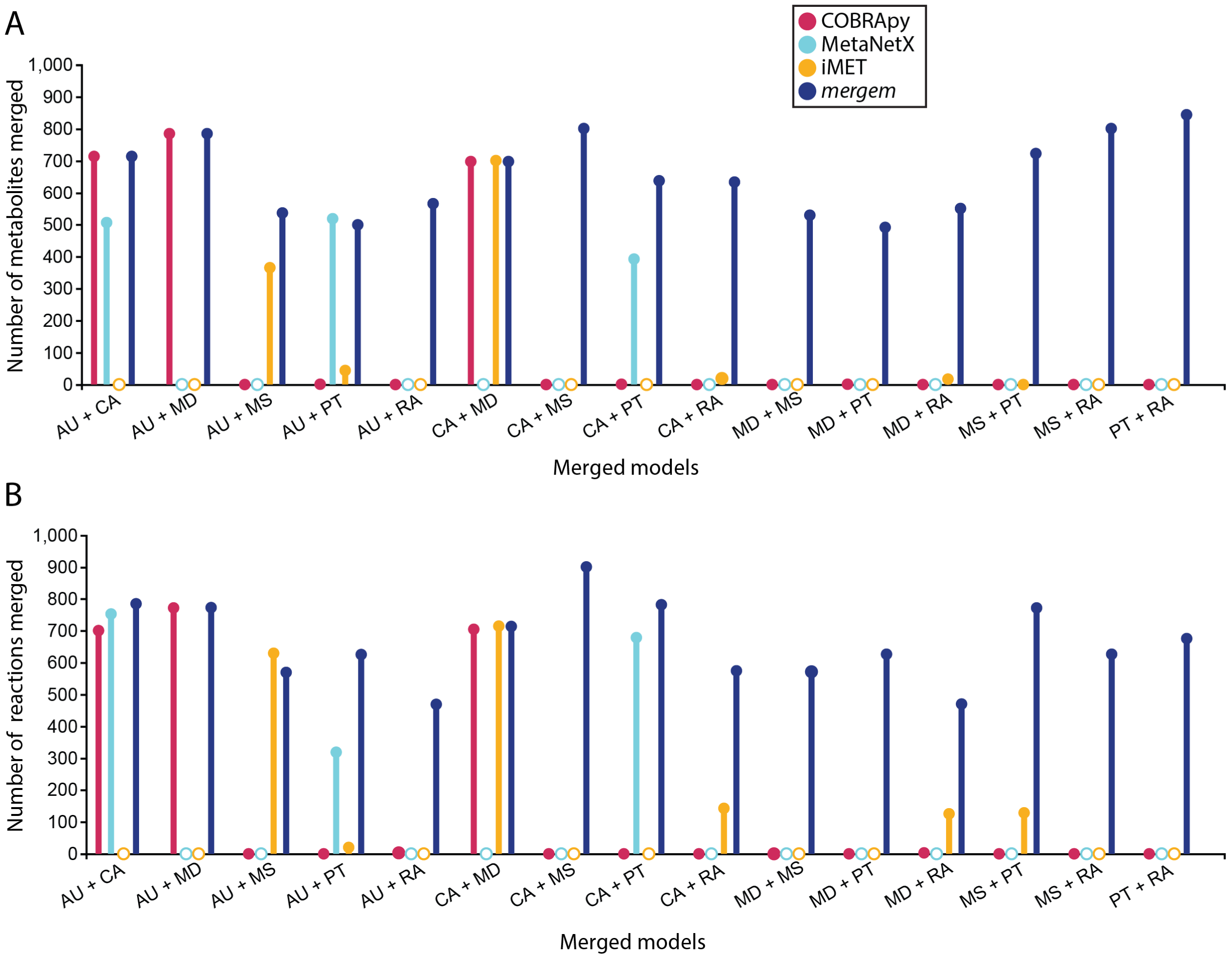


***Figure 4**. Number of metabolites (A) and reactions (B) merged between pairs of six different P. putida reconstructions using four different merging tools: COBRApy, MetaNetX, iMET, and mergem. Filled circles represent successful merging while empty circles indicate failure to load or merge the models. Each reconstruction was drafted with a different pipeline. AU: AuReMe, CA: CarveMe, MS: ModelSEED, MD: MetaDraft, PT: Pathway Tools, and RA: RAVEN.*

2.5. Visually comparing GEMs with *mergem* and Fluxer

Differences between metabolic models can arise not only from the use of different reconstruction pipelines but also from different algorithms used for refinement, as well as from models that represent different biological systems. Comparing such models can help study which metabolites and reactions are unique to each model and common between models. The *mergem* algorithm keeps track of the presence of each metabolite and reaction in each input model. Crucially, this information can aid in the investigation of metabolic differences between the cells or organisms that each model represents and the metabolic coverage in each reconstruction. A graphical visualization of the underlying metabolic networks can further aid in the differential analysis of GEMs.

Fluxer web interface for *mergem* is an ideal tool for visually comparing and analyzing GEMs. The flux network for a merged model can be viewed in the interactive interface as a spanning tree, dendrogram, or complete graph using different layouts. To illustrate this approach for gaining insights regarding metabolic differences, Figure 5 shows the resultant model from merging a platelet (iAT_PLT_636 [30]) and a red blood cell (iAB_RBC_283 [29]) GEM with *mergem* in Fluxer. The nodes and edges in red and blue are unique to the platelet and red blood cell models, respectively. The visualization clearly highlights the different pathways between the two models, such as in the sphingosine and ethanolamine metabolism.

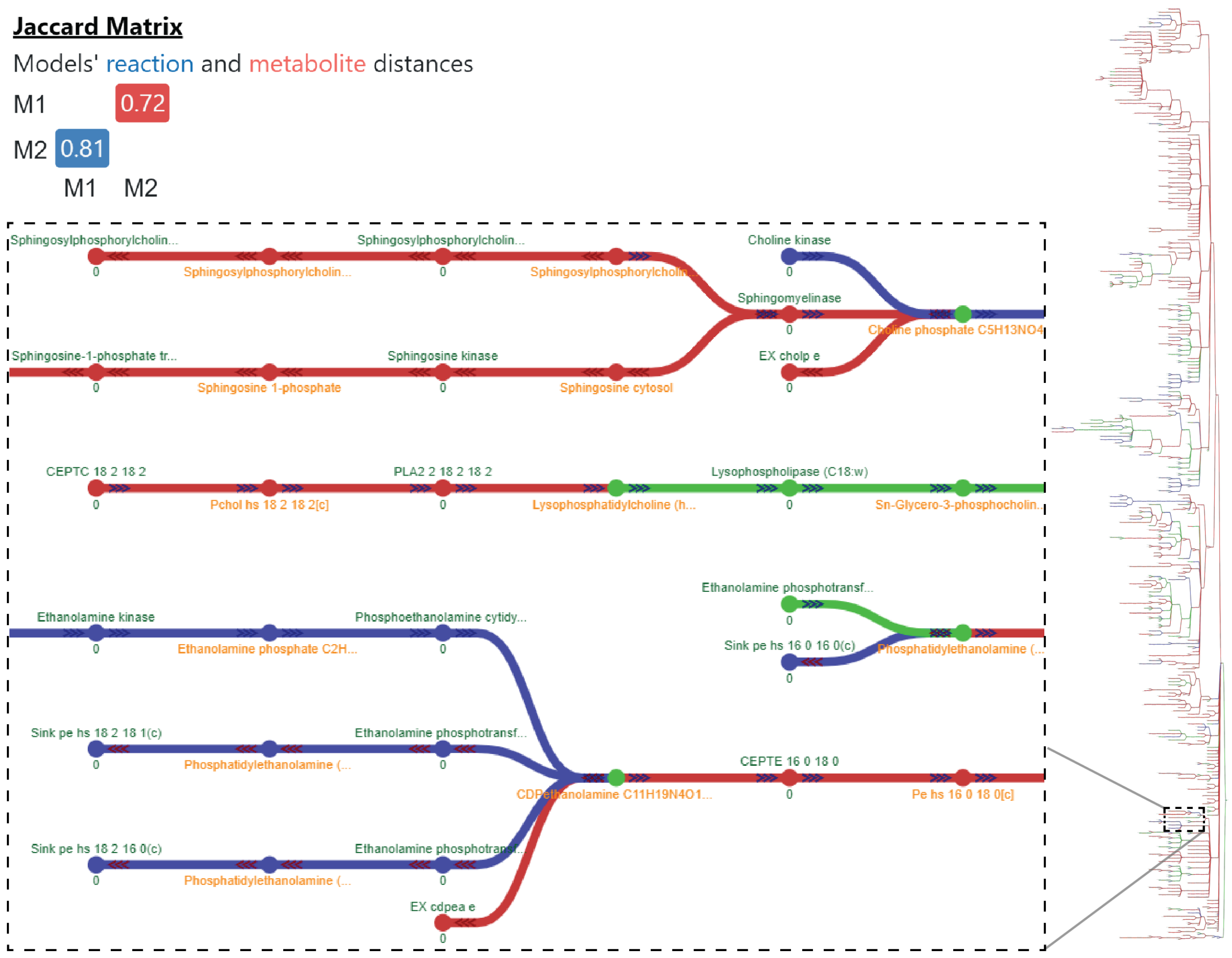

Jaccard Matrix
Models' reaction and metabolite distances
M1 0.72
M2 0.81
M1 M2
Sphingosylphosphorylcholin...
Sphingosylphosphorylcholin...
Sphingosylphosphorylcholin...
Sphingosylphosphorylcholin...
Choline kinase
Sphingomyelinase
Choline phosphate C5H13NO4
Sphingosine-1-phosphate tr...
Sphingosine 1-phosphate
Sphingosine kinase
Sphingosine cytosol
EX cholp e
CEPTC 18 2 18 2
Pchol hs 18 2 18 2[c]
PLA2 2 18 2 18 2
Lysophosphatidylcholine (h...
Lysophospholipase (C18:w)
Sn-Glycero-3-phosphocholin...
Ethanolamine phosphotransf...
Sink pe hs 16 0 16 0(c)
Phosphatidylethanolamine (...
Ethanolamine kinase
Ethanolamine phosphate C2H...
Phosphoethanolamine cytidy...
Sink pe hs 18 2 18 1(c)
Phosphatidylethanolamine (...
Ethanolamine phosphotransf...
CDPethanolamine C11H19N4O1...
CEPTE 16 0 18 0
Pe hs 16 0 18 0[c]
Sink pe hs 18 2 16 0(c)
Phosphatidylethanolamine (...
Ethanolamine phosphotrans...
EX cdpea e

Figure 5. *Visual comparison of GEMs for platelets iAT_PLT_636 and red blood cells iAB_RBC_283 with mergem in Fluxer. The results reveal 72% metabolic and 81% reaction differences between the two models, as shown in the Jaccard matrix. The reaction differences include sphingosine and ethanolamine pathways (inset graph). Nodes and edges in blue are unique to red blood cells while those in red are unique to platelet cells. Elements in green are common to both cell types.*

2.6. Comparing reconstructions

Draft models can vary depending on the pipeline and specific settings used during their reconstructions. Identifying the metabolite and reaction components unique to differently generated reconstructions can aid in curating which components to keep in the final model. There is thus a need for comparing the differences between reconstructions from different pipelines and the integration of the different draft models into a single curated final model. Here we illustrate how *mergem* and Fluxer visualizations can be applied to highlight common and unique elements between draft models.

At the first stage for curating a new GEM, the contents of a draft reconstruction rely on multiple parameters and input data such as the genome sequence, the template model, or gap-filling media. To illustrate how the selection of a different template for a reconstruction can affect the metabolites and reactions added by a pipeline, we compared two *Pseudomonas putida* reconstructions drafted with ModelSEED. The models were generated using either a gram-negative template (MS1) or a core template (MS2) [37]. Figure 6 shows the resulting metabolic spanning tree, without zero-flux reactions and cofactor metabolite nodes, after merging and comparing the two draft models with *mergem* in Fluxer. *mergem* found and merged 1730 metabolites (97.5%) and 1666 reactions (95%) in common between the two reconstructions. The analysis revealed that only the model using MS1 contains fatty acid reactions (Figure 6 inset, red nodes) downstream of the acyl carrier protein. These reactions appear to be associated with the metabolism of lipid poysaccharides that are characteristic of gram-negative bacteria for the synthesis of the lipid bilayer [38]. The components from gram-negative template are thus evident when comparing the two reconstructions that use different templates.

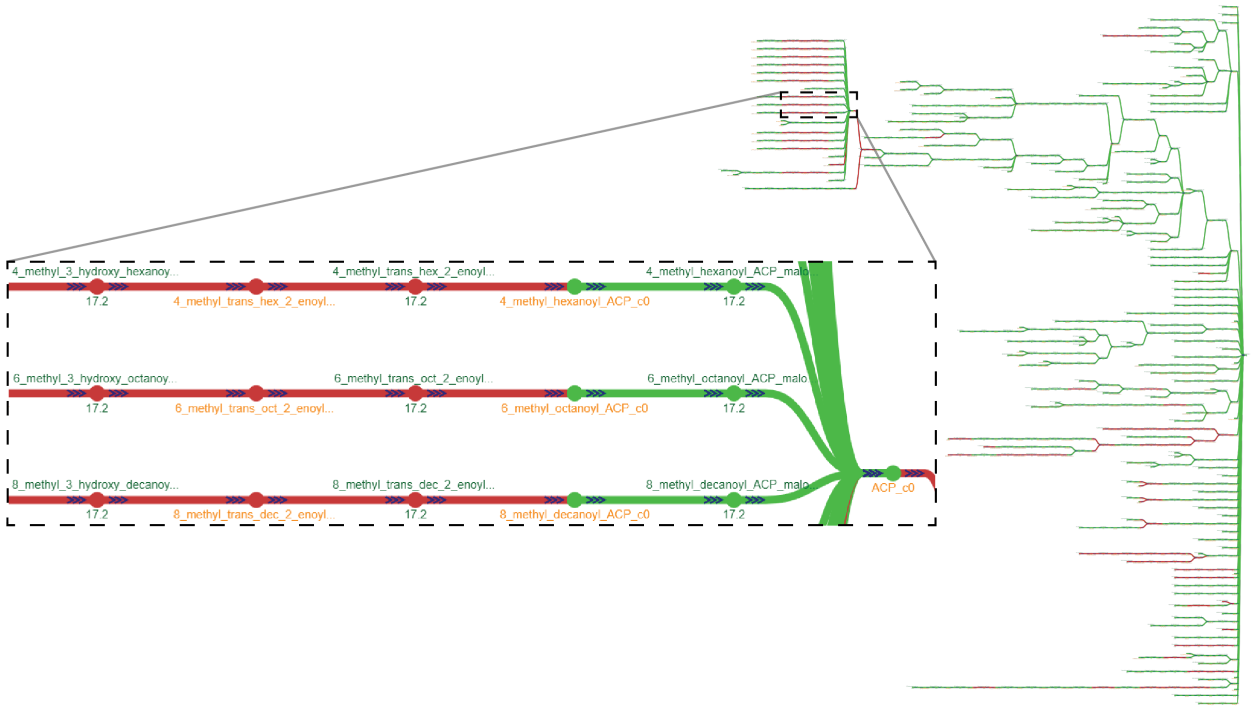

4_methyl_3_hydroxy_hexanoy...
4_methyl_trans_hex_2_enoyl...
4_methyl_hexanoyl_ACP_malo...
17.2
4_methyl_trans_hex_2_enoyl...
17.2
4_methyl_hexanoyl_ACP_c0
17.2
6_methyl_3_hydroxy_octanoy...
6_methyl_trans_oct_2_enoyl...
6_methyl_octanoyl_ACP_malo...
17.2
6_methyl_trans_oct_2_enoyl...
17.2
6_methyl_octanoyl_ACP_c0
17.2
8_methyl_3_hydroxy_decanoy...
8_methyl_trans_dec_2_enoyl...
8_methyl_decanoyl_ACP_malo...
17.2
8_methyl_trans_dec_2_enoyl...
17.2
8_methyl_decanoyl_ACP_c0
17.2
ACP_c0

Figure 6. *Comparing the effect of reconstruction parameters with mergem in Fluxer. Two different P. putida draft reconstructions built with ModelSEED using either a gram-negative (MS1) or core (MS2) template on ModelSEED reveals components unique to gram-negative template-based models. Flux graph of the complete model visualized as a spanning tree. Green nodes and links indicate metabolic components common to both reconstructions and those in red are unique to the gram-negative template reconstruction. Inset shows some of the unique metabolites and reactions involved in the lipopolysaccharide metabolic pathway characteristic of gram-negative bacteria.*

The role of template selection in the CarveMe pipeline was similarly assessed using *mergem* in Fluxer. Two *Lactobacillus plantarum* reconstructions using either a universal bacterial template (CA1) or a gram-positive template (CA2) [37] were merged with *mergem*, resulting in 1031 metabolites (90%) and 1467 reactions (87%) in common between the two reconstructions. Figure 7 shows the resulting metabolic spanning tree, without zero-flux reactions and cofactor metabolite nodes, after merging and comparing the two draft models with Fluxer web interface. The comparison demonstrated that the model from the gram-positive template contained more fatty acid ligase reactions than the one from the universal bacterial template. Additionally, the biomass precursors for the two modes were highly similar (53/61 metabolites in common) except for teichoic acid metabolites, which were unique to the model from the gram-positive template. Indeed, teichoic acid polymers are commonly found in the cell wall of gram-positive bacteria [39,40]. These results highlight the importance of template selection when building draft reconstructions and how visually comparing metabolic networks can be a useful aid for curation.

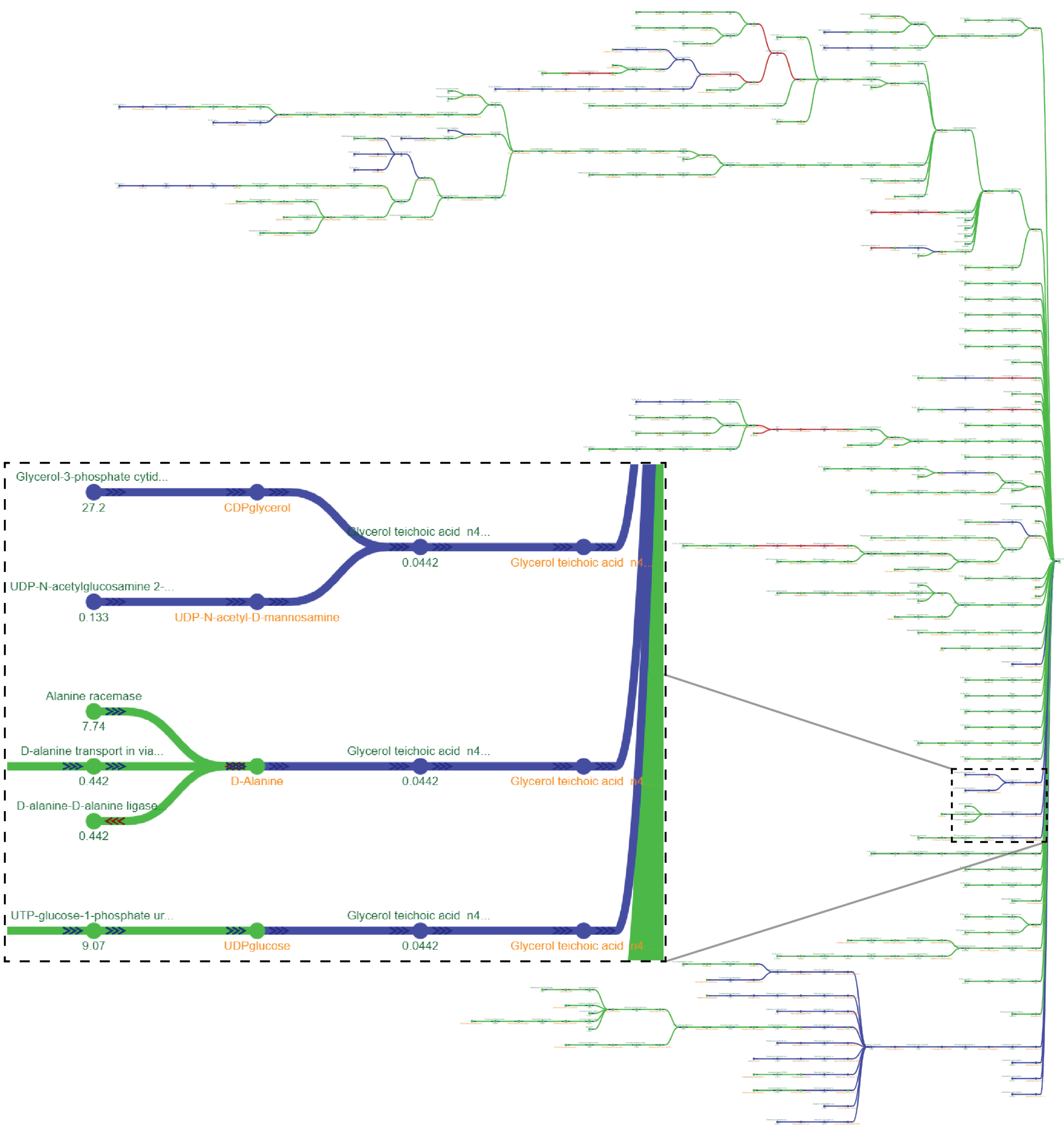
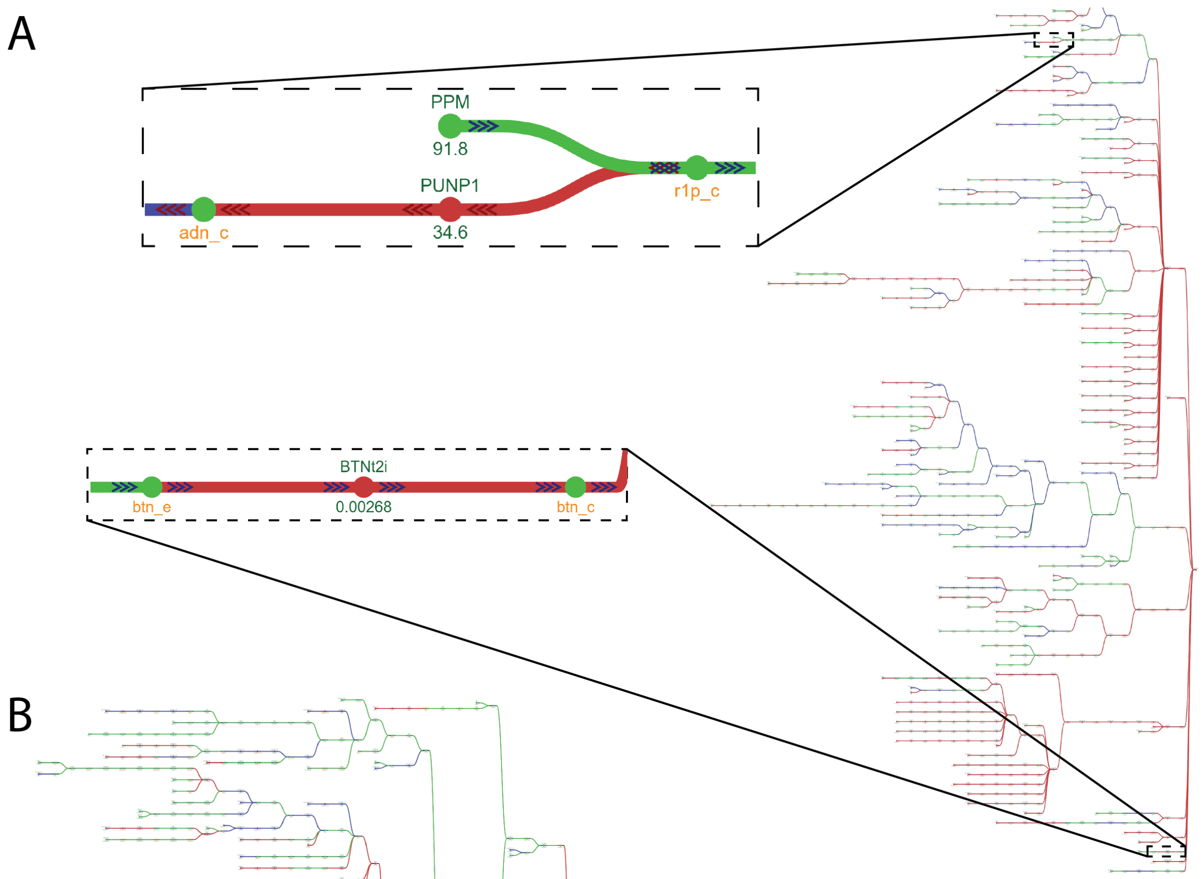

Glycerol-3-phosphate cytid...
27.2
CDPglycerol
Glycerol teichoic acid n4...
0.0442
Glycerol teichoic acid n4...
UDP-N-acetylglucosamine 2-...
0.133
UDP-N-acetyl-D-mannosamine
Alanine racemase
7.74
D-alanine transport in via...
0.442
D-Alanine
Glycerol teichoic acid n4...
0.0442
Glycerol teichoic acid n4
D-alanine-D-alanine ligase
0.442
UTP-glucose-1-phosphate ur...
9.07
UDPglucose
Glycerol teichoic acid n4...
0.0442
Glycerol teichoic acid n4

Figure 7. *Comparing reconstructions using different templates with mergem in Fluxer. Comparing L. plantarum draft reconstructions using a universal (CA1) or gram-positive (CA2) template in CarveMe reveals specific components unique to gram-positive template-based models. Green nodes and links are metabolic components common to both reconstructions and those in red or blue are unique to the universal or gram-positive models, respectively. Blue components include many techoic acids (insert) that is characteristic of gram-positive bacteria.*

2.7. Finding missing and newly-added reactions

Due to incomplete genome annotations—or errors in them—automatically-generated reconstructions can miss essential reactions and hence fail in producing biomass (indicating growth) when FBA-simulated with an appropriate media [41]. A number of tools and algorithms have been developed for finding metabolic and pathway gaps in a reconstruction in base of a set of validated reactions from other organisms. These gap-filling tools include GapFill [42], gapseq [43], and OptFill [44], which Fluxer and *mergem* can complement by visually identifying missing reactions for refining the metabolic model.

Towards the identification of metabolic gaps and possible filling candidates, a draft reconstruction can be compared to a curated model of a closely related organism. To illustrate this approach using *mergem* and Fluxer, a ModelSEED reconstruction for *Lactobacillus plantarum* [37] was merged and compared with a GEM of *Lactobacillus reuteri* [45]. Figure 8A shows the results of merging the two models with *mergem* in Fluxer. The ModelSEED reconstruction shared 409 (25%) metabolites and 436 (26%) reactions with the *L. reuteri* model. Using the Fluxer complete graph, with zero-flux reactions and cofactor metabolites hidden, we identified nine reactions possibly missing in the ModelSEED reconstruction (Supplementary Table 2). To validate if any of these reactions represent true gaps and good gap-filling candidates, we compared the ModelSEED reconstruction with a new curated model of *L. plantarum* [37]. Figure 8B shows the results, revealing that all nine reactions were required to metabolically complete the ModeSEED reconstruction. Inset graphs in Figure 8A and Figure 8B show one of the potential gaps and fillers identified in the analysis. While not all metabolic gaps need to be filled in a final model, *mergem* and Fluxer visualizations represent a crucial aid to improve and assess the completeness of models during curation and refinement.


A
PPM
91.8
PUNP1
34.6
adn_c
r1p_c
BTNt2i
0.00268
btn_e
btn_c

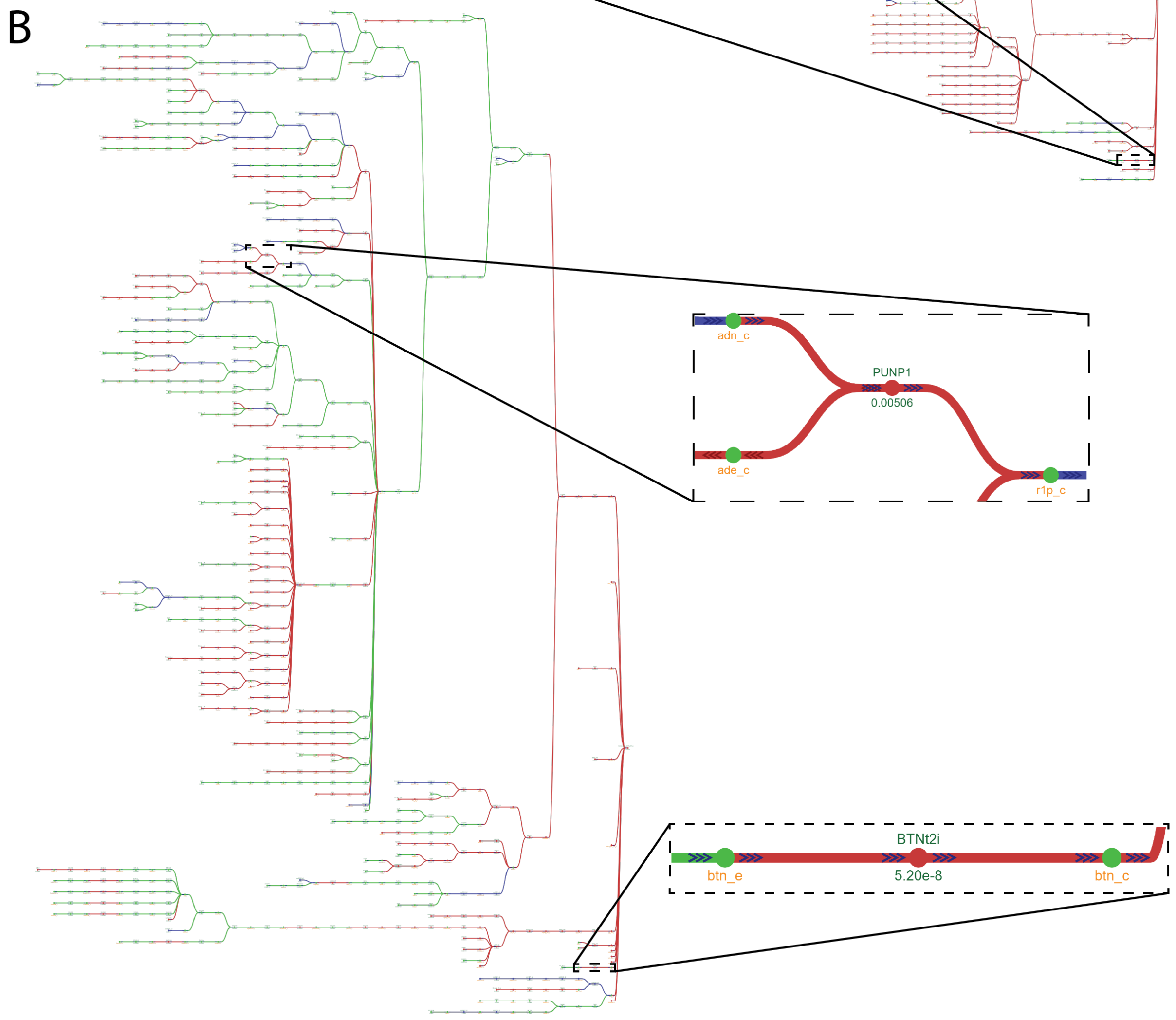
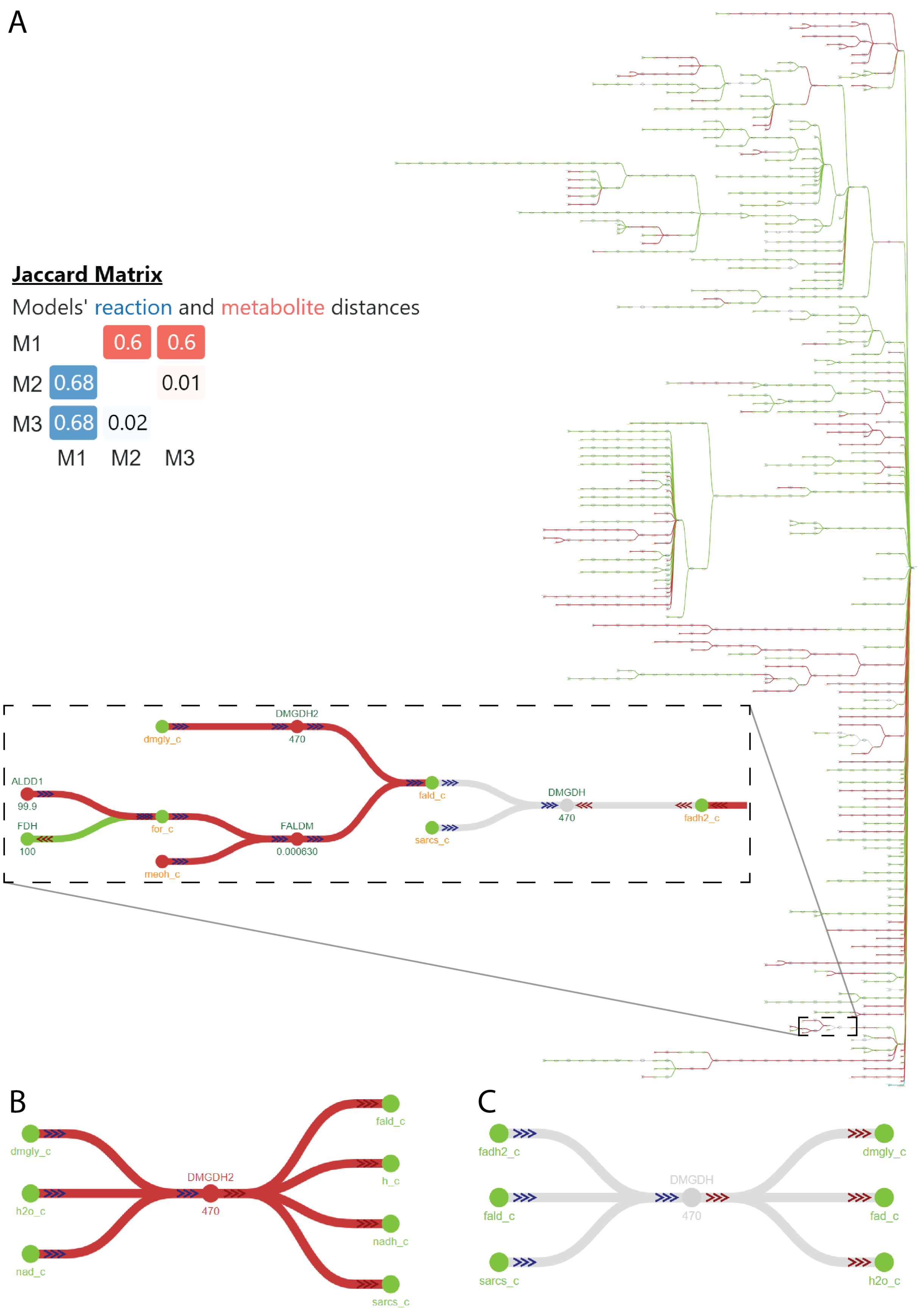

B
adn_c
PUNP1
0.00506
ade_c
r1p_c
BTNt2i
5.20e-8
btn_e
btn_c

Figure 8. *mergem and Fluxer can aid in the identification of possible gaps and filling candidates in draft reconstructions.* ***A.*** *Comparing a ModelSEED draft reconstruction for Lactobacillus plantarum with a manually curated model for Lactobacillus reuteri revealed possible metabolic gaps in the draft model and potential reactions that can fill them. Nodes and links in red are unique to the manually-curated L. reuteri model, while those in green are common between the two models.* ***B.*** *Comparing the same ModelSEED draft reconstruction for L. plantarum with a manually curated model for the organism showed the same reactions as missing in the reconstruction, suggesting that the proposed gaps and fills were correct. The nodes and links in red are unique to the manually curated model and those in green are common between the draft and curated models. Paths zoomed in the inset boxes contain examples of gaps and its filling candidates (Purine nucleoside phosphorylase, PUNP1 and Biotin exchange, BTNt2i) found with this approach.*

Tools and algorithms for gap-filling and other refinement methods most often output the updated models without explicitly listing the newly added reactions and metabolites. In addition, published GEMs are regularly updated when more information is available regarding the corresponding biological system. Comparing a model that has undergone refinement with its previous version can provide insights regarding the metabolic elements that were added as part of the refinement process. To demonstrate how *mergem* and Fluxer can be used to identify updates on models, even when using different identifier namespaces, we downloaded, merged, and compared three models of *Pseudomonas putida* KT2440: iJN1463, the most recently published model for this strain, which is a refinement adding and deleting reactions and metabolites across multiple *P. putida* models [46]; iJN746, the first published GEM for this strain [47]; and MNX_iJN746, a translation of iJN746 to the MetaNetX namespace as retrieved from MetaNetX webserver [12]. Figure 9A shows the resultant comparison by *mergem* with a flux graph in Fluxer, together with its computed Jaccard matrix. The results show how the original model and the version published in MetaNetX, based on a different identifier namespace, are highly similar (99% metabolite match and 99.8% reaction match), but they both differ in 60% and 68% of metabolites and reactions, respectively, with respect to the refined model. This shows how *mergem* can successfully merge and compare models in different database namespaces and highlight differences between model refinements. Aided by the visual comparison in Fluxer, we easily identified that the elements added to the refined model included fatty acids pathways, pyrroloquinoline quinone synthesis, and iron metabolism, as reported in [46]. The inset graph in Figure 9A highlights in red, as visualized in Fluxer, some of the newly-added reactions and metabolites in the refined model. The nodes and links in gray (such as dimethylglycine dehydrogenase reaction, DMGDH) that are common only to the *P. putida* models prior to refinement (both original, iJN746 and its MetaNetX conversion, MNX_iJN746) indicate their deletion or update as part of the model refinement. A closer look at the dimethylglycine dehydrogenase reactions revealed that the older versions of *P. putida* GEMs did not contain nicotinamide dehydrogenase reduction as part of the reaction while the same reaction in the most recent model has been updated to include this cofactor conversion. Similar visual comparisons can be performed with *mergem* for reconstructions before and after gap-filling reactions are applied or for reconstructions obtained from different gap-filling algorithms.



Jaccard Matrix

Models' reaction and metabolite distances

M1		0.6	0.6
M2	0.68		0.01
M3	0.68	0.02	
	M1	M2	M3

Figure 9. *Model updates in terms of metabolic components added, deleted, and retained can be analyzed with mergem and Fluxer. The graphs show the updates performed during the refinements for three P. putida KT2400 models: the most recent model iJN1463, the first published model iJN746 and the first published model in MetaNetX namespace MNX_iJN746.* ***A.*** *Spanning tree graph of merged model and its resultant Jaccard matrix (M1: iJN1463, M2: iJN746, and M3: MNX_iJN746). Graph components in green are common to all models while those in red and purple are unique to the most recent (iJN1463) and first published (iJN746) models, respectively. Inset graph shows examples of metabolites and reactions added (e.g. methanol, meoh_c and formaldehyde dismutase, FALDM), deleted (e.g. dimethylglycine dehydrogenase, DMGDH) and retained (e.g. formate, fom_c, and formate dehydrogenase, FDH). Added, deleted, and retained components are visible in red, gray, and green, respectively.* ***B.*** *The reaction dimethylglycine dehydrogenase was updated in the most recent model to include the reduction of nicotinamide adenine dinucleotide.* ***C.*** *In comparison, the reaction dimethylglycine dehydrogenase did not contain nicotinamide adenine dinucleotide in the previous versions of P. putida GEMs.*

2.8. Comparing closely-related organisms

Organisms of different species but same genus often show many similar but a few unique metabolic phenotypes. Using *mergem* with GEMs of such organisms can aid in globally comparing the metabolisms between related species. To illustrate this case, we compared a GEM for *Pseudomonas aeruginosa* iPau21 [48] with two GEM versions of *Pseudomonas putida* (iJN1463 [46] and iJN746 [47]). iJN746 is the first published model for *P. putida, while* iJN1463 is a refinement of iJN746 and thus contains updated reaction and metabolite information for *P. putida.* Figure 10A shows the complete tree graph for the resultant merged model in radial layout as visualized in Fluxer and the Jaccard matrix resulting from *mergem* comparison. As expected, the two models for *P. putida* (iJN1463 and iJN746) were more similar to each other than to the model for a different species (iPau21). The results further show that the *P. aeruginosa* model is more similar to the older *P. putida* model (iJN746) than its refined counterpart (iJN1463). One cause for this could be the lack of species-specific information at the time of construction of iJN746. Of the 116 biomass precursors in the merged model, 67 (57%) were common between all models, 7 (0.06%) were unique to *P. aeruginosa*, and 18 (15%) were unique to the refined *P.*

putida model, iJN1463. None of the precursors were unique to the *P. putida* model prior the refinement, iJN746. The biomass precursors unique to *P. aeruginosa* included *ubiquinol-9, protein, RNA, Pseudomonas LPS core*, and *peptidoglycan polymer*. Although *P. putida* and *P. aeruginosa* belong to the same genus, they show differences in certain pathways. For example, *P. aeruginosa* contains pathways for production of virulence factors that include alginate (Figure 10B, red elements), rahmnolipids, and phenezines. In the merged model network, the reactions unique to *P. aeruginosa* were primarily lipid reactions (Figure 10C, red elements). Similarly, *P. putida* contains pathways metabolizing aromatic compounds such as toluene, indole, and m-xylene (Figure 10D, gray and purple elements). Similar analyses can be performed to distinguish metabolic features of different organisms.

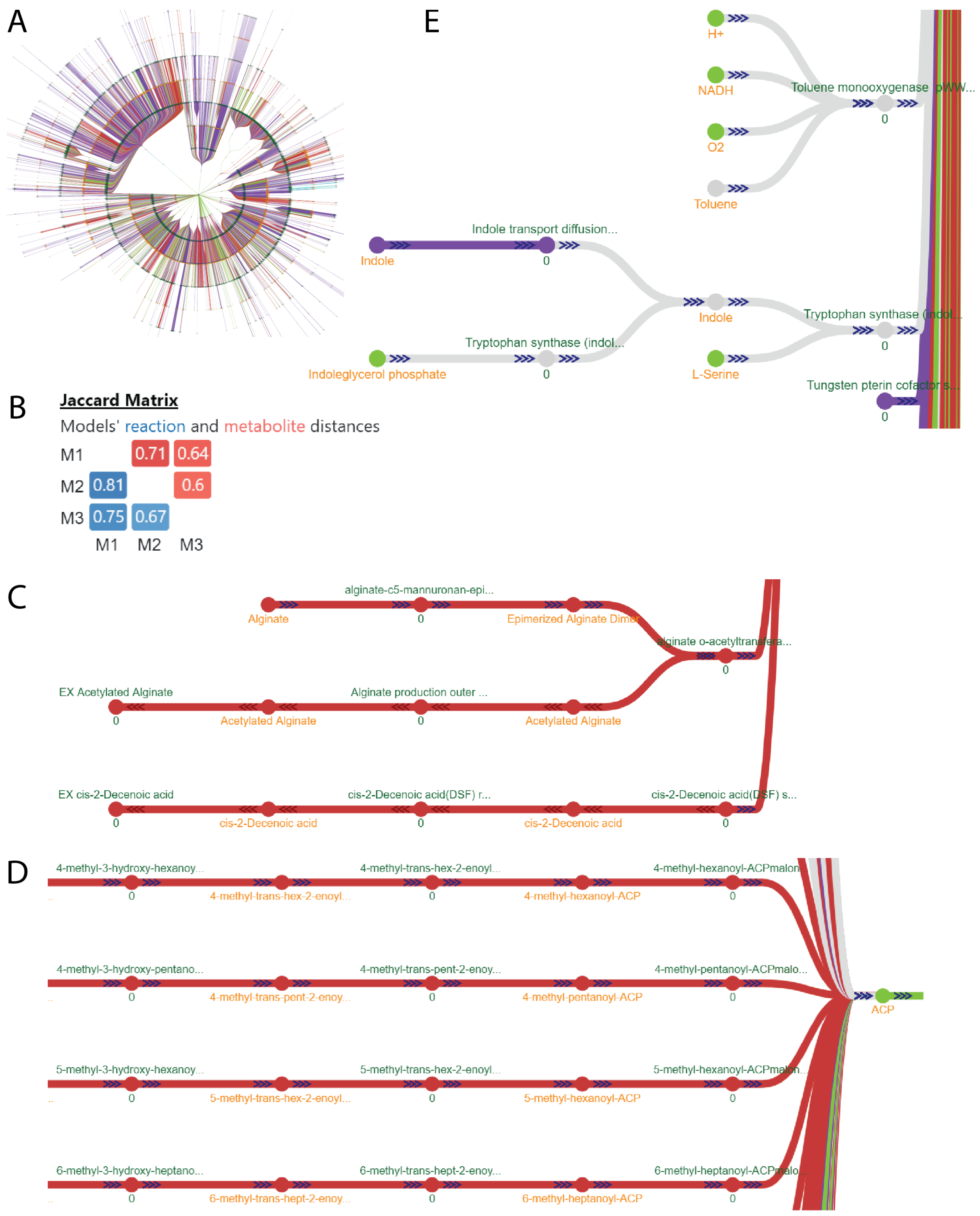

A
E
H+
NADH
Toluene monooxygenase (pWW...
0
O2
Toluene
Indole transport diffusion...
Indole
0
Indole
Tryptophan synthase (indol...
0
Tryptophan synthase (indol...
Indoleglycerol phosphate
0
L-Serine
Tungsten pterin cofactor s...
0
B
Jaccard Matrix
Models' reaction and metabolite distances
M1
M2
M3
0.71
0.64
0.81
0.6
0.75
0.67
C
alginate-c5-mannuronan-epi...
Alginate
0
Epimerized Alginate Dimer
alginate o-acetyltransfera...
0
EX Acetylated Alginate
0
Acetylated Alginate
Alginate production outer ...
0
Acetylated Alginate
EX cis-2-Decenoic acid
0
cis-2-Decenoic acid
cis-2-Decenoic acid(DSF) r...
0
cis-2-Decenoic acid
cis-2-Decenoic acid(DSF) s...
0
D
4-methyl-3-hydroxy-hexanoy...
4-methyl-trans-hex-2-enoyl...
4-methyl-hexanoyl-ACPmalon...
4-methyl-trans-hex-2-enoyl...
4-methyl-hexanoyl-ACP
4-methyl-3-hydroxy-pentano...
4-methyl-trans-pent-2-enoy...
4-methyl-pentanoyl-ACPmalo...
4-methyl-trans-pent-2-enoy...
4-methyl-pentanoyl-ACP
ACP
5-methyl-3-hydroxy-hexanoy...
5-methyl-trans-hex-2-enoyl...
5-methyl-hexanoyl-ACPmalon...
5-methyl-trans-hex-2-enoyl...
5-methyl-hexanoyl-ACP
6-methyl-3-hydroxy-heptano...
6-methyl-trans-hept-2-enoy...
6-methyl-heptanoyl-ACPmalo...
6-methyl-trans-hept-2-enoy...
6-methyl-heptanoyl-ACP

Figure 10*. The metabolism of different organisms can be contrasted by merging and comparing their GEMs with mergem in Fluxer. A comparison of P. aeruginosa (model iPau21) with two models of P. putida (iJN1462 and its previous version as iJN746) reveals their common and* *unique pathways.* ***A.*** *Complete graph of the merged model in radial layout and the resulting Jaccard distance matrix (M1: iPau21, M2: iJN1462, and M3: iJN746).* ***B.*** *Alginate pathway reactions found to be unique to P. aeruginosa.* ***C.*** *Some of the fatty acid pathway reactions found to be unique to P. aeruginosa.* ***D.*** *Toluene and indole and their associated reactions in gray or purple are present only in the P. putida models. Elements in green are common to all models; elements in red are unique to P. aeruginosa; elements in purple are unique to iJN1462; elements in gray are common to two models.*

2.9. Comparing organisms for finding novel drug targets

Comparing metabolic networks to find commonalities can facilitate the discovery of potential broadly-distributed targets, such as for developing new antimicrobial drugs [49]. To illustrate this application, we used *mergem* in Fluxer to visualize the commonalities between the curated GEMs for three gram-negative pathogens: *Acinetobacter baumanii* [50], *Klebsiella pneumoniae* [51], and *Pseudomonas aeruginosa* [48]. Figure 11 shows the visualization of the resultant merged models in Fluxer. Since the three gram-negative pathogens contained around 60% unique reactions and 70% unique metabolites, as shown in the Jaccard matrix (Figure 11), we hypothesized that pathways common to all three pathogens could be candidates for antimicrobial drugs. Among the common elements, this comparison revealed the shikimate and riboflavin pathways (Figure 11, green), which are indeed targets of current antimicrobials [52,53]. The other reactions found in common are good candidates for further study to test their effectiveness as antibacterial targets against gram-negative bacterial pathogens.

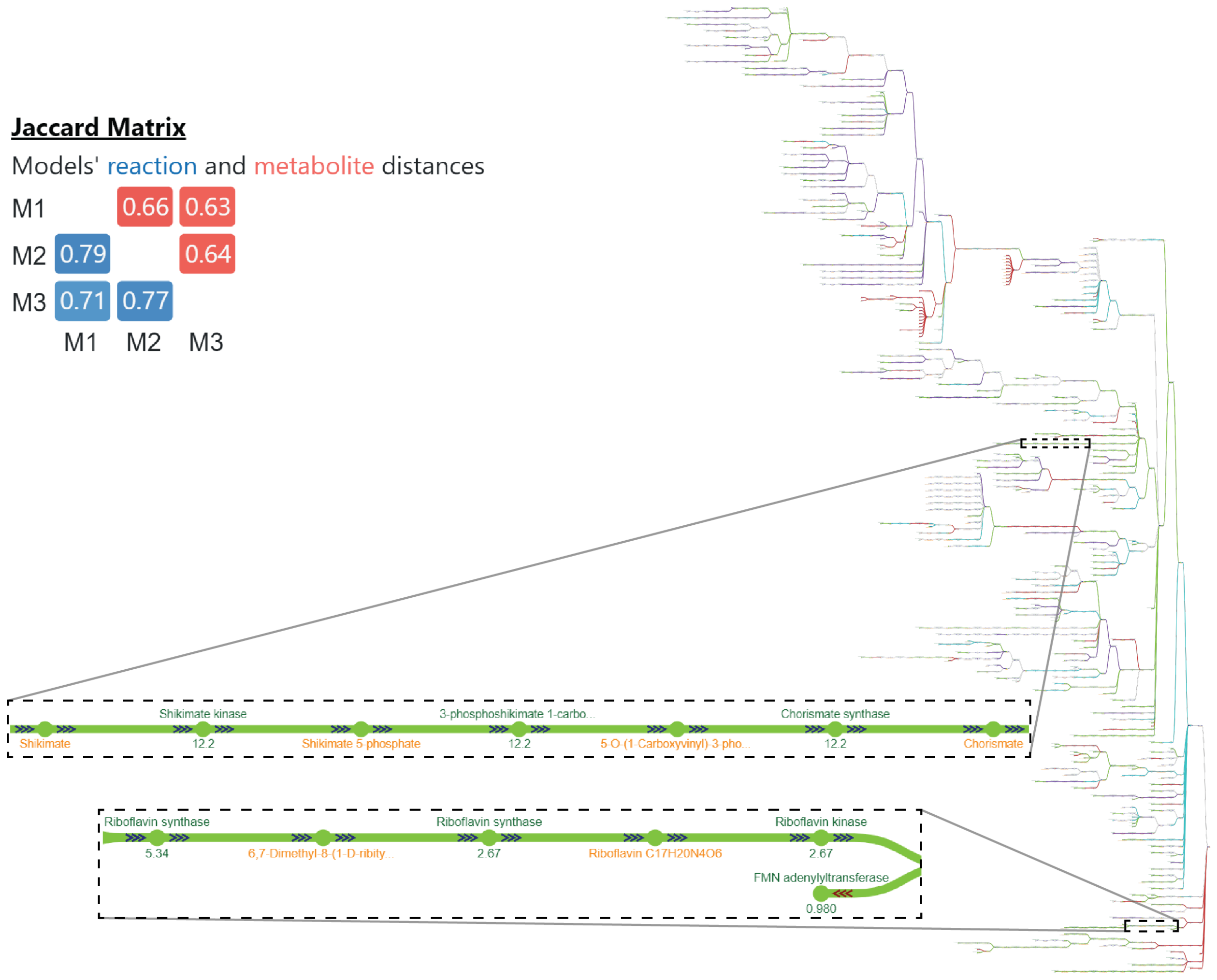


Figure 11. *Potential antibacterial targets were identified with mergem in Fluxer by comparing GEMs of three gram-negative pathogens, including Acinetobacter baumanii, Klebsiella pneumoniae, and Pseudomonas aeruginosa. The resulting common pathways included shikimate and riboflavin (insets), which hence are good candidates as antimicrobial targets.*

3. Discussion

Here we presented *mergem*, a novel tool for merging and comparing GEMs. The method computes and uses a universal database identifier mapping dictionary to translate metabolite ids from multiple different database systems to a common namespace. This allows reactions to be merged by comparing the translated metabolite IDs of their reactants and products. *mergem* can be used on the command-line or imported into Python scripts. While there exist other tools that can merge GEMs, *mergem* shows improvement over these methods in terms of recognition and reconciliation of metabolite identifiers, merging of reactions based on these metabolite mappings, and range of model inputs.

To enable the visual and interactive comparison of models and the construction of merged models with a user-friendly tool, we incorporated the presented methodology into the web-application Fluxer. This approach can create, visualize, and simulate interactive flux graphs of multiple merged models in which metabolites and reactions are color-coded depending on their model source. We demonstrated this visual methodology for merging and comparing draft reconstructions from different pipelines, even when the models use different database identifiers. In addition, we showed applications of the methodology for visually identifying potential gaps in reconstructions and for finding commonalities useful for discovering antimicrobial targets.

Discrepancies between GEMs and reconstructions for the same organism or cellular system can arise from any of the steps during curation: (1) genome annotation, (2) environment specification, (3) biomass formulation, (4) network gap-filling, and (5) flux simulation method [54]. Merging and visually comparing the metabolic models at each stage of the reconstruction process using the proposed methodology with *mergem* and Fluxer, could help in the detection of erroneous or missing metabolic data in the models. The examples described illustrate how *mergem* can aid in resolving these five sources of discrepancies, from finding missing reactions by comparing close-related organisms to validating a curated model with a FROG report using different simulation methods. Future work will extend the *mergem* algorithm to include gene expression, metabolomic, and fluxomic data for comparing metabolic networks. Such extensions will further increase our ability for visualizing and contrasting different metabolic phenotypes.

GEMs are essential tools for understanding and predicting metabolic behaviors in research and engineering applications. The process of building genome-scale models involves working with many drafts and refining them iteratively. Unfortunately, different reconstruction tools use different database identifiers, which makes it challenging to compare and integrate models from different pipelines. While there is no gold standard pipeline for generating comprehensive drafts, we showed that using *mergem* to integrate drafts from multiple tools could be a valuable aid for model curation. While *mergem* represents a significant advancement for accurately mapping identifiers from several databases, there is still a need for a global standardization in metabolite and reaction identifiers and definitions. Such standardization certainly would improve merging tools such as *mergem* and facilitate the curation of comprehensive GEMs and their comparison.

4. Conclusions

mergem and its interactive visual integration in Fluxer enable new comparative studies of GEMs for different reconstructions, curated models, and organisms. In addition to providing a means to construct standardized and comprehensive GEMs, the presented tools will be an aid for generating hypotheses regarding the genetic targets that can be overexpressed or knocked-out to optimize any phenotype, such as those in disease-relevant pathways or for the production of value-added metabolites in engineering. Future applications will include using *mergem* and Fluxer to build and analyze robust GEMs models that can predict optimized growth phenotypes to be further integrated with mechanistic models and machine-learning methodologies [55–58].

5. Methods

5.1. Jaccard Distance between GEMs

The differences between two GEMs in terms of common metabolites and reactions can be quantified using the Jaccard distance as

$$\mathrm{J}(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

where, A and B represent either the metabolite or reaction sets for each of the models. The Jaccard distance range is [0,1], where 0 indicates that the sets (metabolites or reactions) are identical and 1 indicates that none of the elements are matched between the models.

Since the Jaccard distance is commutative and equal to zero when applied to the same set, we define a Jaccard distance matrix JM for a set of n GEMs as

$$\mathrm{JM}(G_1, \dots, G_n) = \begin{bmatrix} 0 & \mathrm{J}(M_1, M_2) & \dots & \mathrm{J}(M_1, M_n) \\ \mathrm{J}(R_2, R_1) & 0 & \ddots & \vdots \\ \vdots & \ddots & 0 & \mathrm{J}(M_n, M_n) \\ \mathrm{J}(R_n, R_1) & \dots & \mathrm{J}(R_n, R_n) & 0 \end{bmatrix}$$

where M_i and R_i denote the set of metabolites and reactions, respectively, in G_i.

5.2. Algorithm for merging models

The *mergem* algorithm for merging two or more GEMs can be summarized into five major steps:

1. Template model assignment
2. Merging subsequent input models
3. Assigning objective function to merged model
4. Calculating Jaccard distances between pairs of models
5. Returning merged model, metabolite and reaction mappings, and Jaccard distance matrix

5.2.1. Definitions

GEM

A GEM G is a system (M_G, R_G, O_G), where M_G is a set of metabolites, R_G is a set of metabolic reactions, and O_G is a set of objective reactions in G. A reaction r is an ordered pair (T_r, P_r), where T_r is a set of reactant metabolites and P_r is a set of product metabolites. $S(m, T_r)$ and $S(m, P_r)$ denote the stoichiometry of metabolite m in reactants T_r or products P_r, respectively.

mergem identifier

A *mergem* ID is a string formed with the concatenation of a universal identifier (from the mapping system derived in Section 2.1) and the subsystem that the molecule is found in (*e.g.*, cytoplasm, extracellular, mitochondria). For example, a hydrogen atom in the cytoplasm is represented as *mergem_78_c*. If no universal identifier exists for the metabolite, its original identifier is used instead. Formally, $\mathrm{ID}(\mathrm{m})$ denotes the identifier of metabolite m and $\mathrm{ID}_\theta(m)$ denotes its mapped *mergem* ID. This naming system is only internal to the algorithm for merging common metabolites from the input models to overcome the problem with models using different namespaces. The output merged model does not contain any *mergem* identifier, since the *mergem* IDs are reverted to the original identifier used by the first input model containing it. The original metabolite identifiers are temporarily stored for reversion as $\sigma: m \rightarrow \mathrm{ID}(\mathrm{m})$.

Reaction key

A reaction key contains the *mergem* IDs of its participating metabolites, each paired with an integer denoting it as a reactant (-1) or product (1) in the reaction. In particular, a reaction key $\mathrm{K}(r)$ of a reaction r is a set of pairs $\{(ID_\theta(m_i), -1), \ldots (ID_\theta(m_j), 1), \ldots\} | m_i \in T_r, m_j \in P_r$. For example, the key for the reaction

$$mergem\_23\_c + mergem\_45\_c \ \rightarrow mergem\_67\_c$$

would be

$$\{(mergem\_23\_c, -1), (mergem\_45\_c, -1), (mergem\_67\_c, 1)\}$$

Reaction databases such as MetaNetX explicitly include hydrogen atoms in reaction products and reactants, but others do not include them. Further, MetaNetX models contain a special model compartment called 'boundary' to represent model boundary that is independent of other commonly found biological compartments in GEMs, such as 'cytoplasm', 'mitochondria', or 'extracellular'. Since these conventions are not standardized across all reaction databases, the *mergem* algorithm is designed to ignore explicit hydrogen atoms and boundary metabolites when generating reaction keys, which allows for better reaction matching across databases.

Source mapping dictionaries

For analyzing the differences and commonalities between the input models, the algorithm tracks the presence of each merged metabolite and reaction in each input model. These mappings are formally defined for metabolites with $\mu: m \in M_\cup \rightarrow \mathrm{G}^* \subseteq \{G_1, \ldots, G_n\}$ and for reactions with $\rho: r \in R_\cup \rightarrow \mathrm{G}^* \subseteq \{G_1, \ldots, G_n\}$.

Algorithm input

A tuple of n GEMs $[G_1, \ldots, G_n]$ to merge and whether the objective should be merged or selected.

Algorithm output

The merged GEM $G_\cup$, a source mapping of metabolites in the merged model to their original models $\mu: m \in M_\cup \rightarrow \mathrm{G}^* \subseteq \{G_1, \ldots, G_n\}$, a source mapping of reactions in the merged model to their original models $\rho: r \in R_\cup \rightarrow \mathrm{G}^* \subseteq \{G_1, \ldots, G_n\}$, and a Jaccard distance matrix JM with all pair-wise metabolite and reaction distances between the input models.

5.2.2. Assigning and translating the template model

The *mergem* algorithm begins by initializing the first input model as template for merging.

Template metabolites

Each metabolite in the template model is translated into its corresponding *mergem* ID and copied to the merged model. Each translation is performed only after checking that a metabolite with the same *mergem* ID is not already present in the merged model. This can happen when the model contains different metabolites that map to the same *mergem* ID. For example, NH2 and NH3 map to the same *mergem* ID since many databases consider them synonymous due to their difference in a single proton. In these cases, the original metabolite identifier is retained and not translated. This guarantees that the merged model contains all the metabolites from the template (first) model and that the rest of the models can merge the same conflicting metabolites to the template model correctly.

Template reactions

All the template model reactions are copied to the merged model, and their reaction keys generated for comparison with subsequent models. Notice that reaction keys always contain the translated *mergem* IDs, independently if different metabolites are translated to the same *mergem* ID. This maximizes the ability of the algorithm to merge equivalent reactions.

5.2.3. Merging models

Subsequent input models are then merged into the merged model by translating their metabolites to *mergem* IDs and generating keys for their reactions.

Metabolite translation

Metabolites in the non-template models are translated to *mergem* IDs, except when their original IDs are not found in the universal ID mapping system or it already exists in the merged model—which can occur when multiple metabolites in the template model map to the same *mergem* ID. In these cases, their original ID is maintained according to the priority established by the order of the input models. Similar to the template model, different metabolites in a model to merge can map to the same *mergem* ID. In this case, the reactions in the model to merge are updated with the metabolite labeled with the new *mergem* ID. An exception to this rule is when the same reaction contains both metabolites and such substitution would affect the stoichiometry (or even remove the metabolite altogether from the reaction if their stoichiometries cancel out). In this case, the metabolite original ID is maintained. This methodology results in maximizing correct reaction matchings while conserving original reactions when conflicts appear in translated metabolite IDs.

Reaction matching and merging

Reactions not in the model objective function (which are processed separately) are merged by adding new reactions to the merged model only when their reaction key does not exist among the reactions already included in the merged model. Both the forward and reverse keys are considered a match, since their bounds define the actual flux direction in a model. Reaction IDs are not changed during the merging process, except when another reaction with the same ID already exists in the merged model. In this case, the symbol '~' is appended to the identifier of the new reaction to avoid the ID conflict.

5.2.4. Assigning an objective function

After all the input models have been merged, an objective function is added to the merged model according to the criteria selected by the user: either a single objective function form an input model or a merged objective function among all the input models.

Single objective function

If the user selected an objective function from one of the input models, all objective reactions from that model are added to the merged model and the merged model is assigned the same objective function.

Merged objective function

If the user selected to merge all objective functions, all the reactions in the objective functions from all the input models are merged into a single reaction. Metabolites included in multiple reactions are included only once in the merged objective reaction and its stoichiometry is averaged among all the objective reactions containing it. This option is particularly useful for merging and comparing biomass functions and compositions between different models.

5.3. Algorithm pseudocode

// Step 1. Assign template model

```
1.  $G_\cup \leftarrow (\emptyset, \emptyset, \emptyset)$                         // Initialize merged model as empty
2.  $\forall m \in M_{G_1}$:                                   // For each metabolite in template model
3.      $M_{G_\cup} \leftarrow M_{G_\cup} \cup \{m\}$                  // Add metabolite to merged model
4.      $\mu(m) \leftarrow \{G_1\}$                            // Update source mapping
5.      if $\mathrm{ID}_\theta(m) <> \emptyset$ AND                    // If the metabolite maps to a universal ID and
6.        $\nexists m' \in M_{G_\cup} \mid \mathrm{ID}_\theta(m') == \mathrm{ID}(m)$:   // no previous metabolite mapped to it
7.          $\sigma(m) \leftarrow \mathrm{ID}(m)$                      // Store original metabolite id
8.          $\mathrm{ID}(m) \leftarrow \mathrm{ID}_\theta(m)$                   // Convert metabolite ID to universal ID
9.  $\forall r \in R_{G_1}$:                                   // For each reaction in template
10.     $R_{G_\cup} \leftarrow R_{G_\cup} \cup \{r\}$                  // Add reaction to merged model
11.     $\Psi(\mathrm{K}(\mathrm{r})) \leftarrow \mathrm{r}$                          // Create and store reaction key
12.     $\rho(r) \leftarrow \{G_1\}$                            // Update source mapping
```

// Step 2. Merge other input models

```
13. $\forall G \in [G_2, \dots, G_n]$:                          // For each non-template model in input
14.     $\forall m \in M_G$:                                // For each metabolite in model
15.         if $\mathrm{ID}_\theta(m) == \emptyset$                     // If the metabolite has no universal ID
16.             $\mu(m) \leftarrow \mu(m) \cup \{G\}$               // Update source mapping
17.             if $\nexists m' \in M_{G_\cup} \mid \mathrm{ID}(m') == \mathrm{ID}(m)$:  // If no previous metabolite exists with same ID
18.                 $M_{G_\cup} \leftarrow M_{G_\cup} \cup \{m\}$          // Add metabolite to merged model
19.         elif $\exists m' \in M_{G_\cup} \mid \mathrm{ID}(m') == \mathrm{ID}(m)$:    // Metabolite exists with same original ID
20.             $\mu(m) \leftarrow \mu(m) \cup \{G\}$               // Update source mapping
21.         elif $\exists m' \in M_{G_\cup} \mid \mathrm{ID}(m') == \mathrm{ID}_\theta(m)$:   // Metabolite exists with same universal ID
22.             if $\mathrm{ID}(m) \notin \sigma(\mathrm{ID}_\theta(m))$ AND          // If its original ID not already mapped but
23.               $\exists m' \in M_G | \mathrm{ID}(m') \in \sigma(\mathrm{ID}_\theta(m))$     // the model has one that could be mapped
24.                 $\mu(m) \leftarrow \mu(m) \cup \{G\}$           // Update source mapping
25.                 $M_{G_\cup} \leftarrow M_{G_\cup} \cup \{m\}$          // Add metabolite to merged model
26.             elif $G \in \mu(m')$:                      // Translated ID already found in current model
27.                 for $r \in R_G | m \in T_r \cup P_r$:          // For each reaction with metabolite
28.                     if $m' \in T_r \cup P_r$:              // Original metabolite ID also in reaction
29.                         $\mu(m) \leftarrow \mu(m) \cup \{G\}$   // Keep original ID and update mapping
30.                         $M_{G_\cup} \leftarrow M_{G_\cup} \cup \{m\}$  // Add metabolite to merged model
31.                     else:
32.                         $T_r \leftarrow T_r \backslash m \rightarrow m'$        // Substitute original for mapped metabolite
33.                         $P_r \leftarrow P_r \backslash m \rightarrow m'$        // in reactant and products
34.             else:
35.                 $\mathrm{ID}(m) \leftarrow \mathrm{ID}_\theta(m)$              // Convert metabolite id with *mergem* ID mapper
36.                 $\mu(m) \leftarrow \mu(m) \cup \{G\}$           // Update source mapping
37.         else:
38.             $\mathrm{ID}(m) \leftarrow \mathrm{ID}_\theta(m)$                  // Convert metabolite id with *mergem* ID mapper
```

```
39.             $\mu(m) \leftarrow \mu(m) \cup \{G\}$                // Update source mapping
40.             $M_{G_\cup} \leftarrow M_{G_\cup} \cup \{m\}$        // Add metabolite to merged model
41.         $\sigma(m) \leftarrow \mathrm{ID}(m)$                    // Store original metabolite id
42.     $\forall r \in R_G$:                                         // For each reaction in model
43.         if $\mathrm{K}(r) \in \Psi$:                             // Reaction key exists
44.             $\rho(\Psi(\mathrm{K}(r))) \leftarrow \rho(\Psi(\mathrm{K}(r))) \cup \{G\}$    // Update source mapping
45.         elif $\mathrm{K}(r)^{-1} \in \Psi$:                      // Inverse reaction key exists
46.             $\rho(\Psi(\mathrm{K}(r)^{-1})) \leftarrow \rho(\Psi(\mathrm{K}(r)^{-1})) \cup \{G\}$ // Update source mapping
47.         else:                                                    // Reaction key does not exist
48.                 $R_{G_\cup} \leftarrow R_{G_\cup} \cup \{r\}$    // Add reaction to merged model
49.                 $\Psi(\mathrm{K}(r)) \leftarrow r$               // Store reaction key
50.                 $\rho(r) \leftarrow \rho(r) \cup \{G\}$          // Update source mapping
```

// Step 3. Assign objective function

```
51. if objective from $G_i$:                         // If objective function is from input GEM i
52.     $\forall r \in O_{G_i}$:                     // For each objective reaction
53.         $\rho(r) \leftarrow \rho(r) \cup \{G_i\}$ // Update source mapping
54.     $O_{G_\cup} \leftarrow O_{G_i}$              // Set it as objective of the merged model
55. else:                                            // The objective functions are merged
56.     $r^* \leftarrow (\emptyset, \emptyset)$     // Initialize objective function reaction as empty
57.     $\forall G \in [G_1, \dots, G_n]$:          // For each input GEM
58.         $\forall r \in O_G$:                     // For each objective reaction
59.             $\rho(r) \leftarrow \rho(r) \cup \{G\}$   // Update source mapping
60.             $\forall m \in T_r$:                 // For each reactant metabolite in reaction
61.                 if $\exists m^* \in T_{r^*} \mid \mathrm{ID}(m^*) == \mathrm{ID}(m)$:   // If reactant already exists in merged obj reac
62.                     $S(m^*, T_{r^*}) \leftarrow \frac{\left(\delta(m^*,T_{r^*}) \cdot S(m^*,T_{r^*}) + S(m,T_r)\right)}{\delta(m^*,T_{r^*})+1}$   // Average the stoichiometry
63.                 else:
64.                     $T_{r^*} \leftarrow T_{r^*} \cup \{m\}$   // Add reactant to merged obj reaction
65.             $\forall m \in P_r$:                 // For each product metabolite in reaction
66.                 if $\exists m^* \in P_{r^*} \mid \mathrm{ID}(m^*) == \mathrm{ID}(m)$:   // If product already exists in merged obj reac
67.                     $S(m^*, P_{r^*}) \leftarrow \frac{\left(\delta(m^*,P_{r^*}) \cdot S(m^*,P_{r^*}) + S(m,P_r)\right)}{\delta(m^*,P_{r^*})+1}$   // Average the stoichiometry
68.                 else:
69.                     $P_{r^*} \leftarrow P_{r^*} \cup \{m\}$   // Add reactant to merged obj reaction
70.     $O_{G_\cup} \leftarrow \{r^*\}$             // Add merged obj reaction as obj function
                                                     // of merged model
```

*// **Step 4: Calculate Jaccard distances***

```
71.     $\forall i, j \in [1, \dots, n]$:            // For each pair of input GEMs
72.         if $i < j$:
73.             $JM_{i,j} \leftarrow 1 - \frac{|M_i \cap M_j|}{|M_i \cup M_j|}$   // Add Jaccard distance to distance matrix
74.         if $i > j$:
75.             $JM_{i,j} \leftarrow 1 - \frac{|R_i \cap R_j|}{|R_i \cup R_j|}$   // Add Jaccard distance to distance matrix
76.         else:
77.             $JM_{i,j} \leftarrow 0$              // Distance matrix diagonal is zero
```

```
// *Step 5: Revert metabolite IDs and return results*
78. $\forall m \in M_{G_\cup}$:                 // For each metabolite in merged model
79.     $\mathrm{ID}(m) \leftarrow \sigma(m)$       // Restore original metabolite id
80. return $(G_\cup, \mu, \rho, JM)$            // Return merged GEM, mappings, and Jaccard
                                         // distance matrix
```

5.4. *mergem* Python package

The merging and comparison algorithm has been implemented as the Python package *mergem*, which is freely available in the Python Package Index, PyPI (https://pypi.org/project/mergem/). The package can be executed on the command-line or imported into Python scripts. Depending on the execution method, users can provide a list of COBRApy model objects or filenames (in SBML, MATLAB, or JSON formats) as input to *mergem* along with the objective function to use. The command-line execution automatically saves the resulting merged model as an SBML file, or any other output filename and format specified by the user for maximum compatibility with other tools. The package also computes and returns the Jaccard distance matrix between the models and dictionaries with the source models for each metabolite and reaction, as well as provides methods to translate metabolite and reaction universal identifiers and retrieve their consolidated properties.

5.5. Fluxer web interface

Fluxer [17] uses HTML5 and JavaScript for the front end and Python and Flask (Pallets) for the backend. Models loaded into the application are stored in an internal SQLite database. The COBRApy [26] package is used to read, FBA-optimize, and write the SBML model files. Fluxer uses the *mergem* package for merging input metabolic networks and identifying unique and common metabolic components within the input models as well as for retrieving their properties. Graph layouts are visualized with the D3.js library [59]. FROG reports are generated with the *fbc_curation* package [31].

6. Declarations

Data Availability

mergem can be freely installed from PyPI with *pip*, the package installer for Python. The source code is available at https://github.com/lobolab/mergem. Detailed documentation for using *mergem* is available at https://mergem.readthedocs.io. All the models shown in the figures are freely- available in Fluxer at https://fluxer.umbc.edu/ and their URLs are listed in Supplementary Table 3.

Competing interests

The authors declare that they have no competing interests.

Funding

This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health under award number R35GM137953. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. AH was supported in part by a fellowship from Merck Sharp & Dohme Corp.

Authors' contributions

AH and DL designed and implemented the method and wrote the manuscript. AH analyzed the data. DL secured funding. All authors read and approved the final manuscript.

Acknowledgements

We thank Daniel Machado for useful discussions, Ivan Erill and Jeffrey Gardner for critical reading of the manuscript, and the members of the Lobo Lab and the students and faculty at UMBC for their support and help in testing the software applications.

References

1. Gu C, Kim GB, Kim WJ, Kim HU, Lee SY. Current status and applications of genome-scale metabolic models. Genome Biology. 2019;20:1–18.

2. Fang X, Lloyd CJ, Palsson BO. Reconstructing organisms in silico: genome-scale models and their emerging applications. Nature Reviews Microbiology. Springer US; 2020;18:731–43.

3. Thiele I, Palsson B. A protocol for generating a high-quality genome-scale metabolic reconstruction. Nature Protocols. Nature Publishing Group; 2010;5:93–121.

4. Monk J, Nogales J, Palsson BO. Optimizing genome-scale network reconstructions. Nature Biotechnology. Nature Publishing Group; 2014;32:447–52.

5. Faria JP, Rocha M, Rocha I, Henry CS. Methods for automated genome-scale metabolic model reconstruction. Biochemical Society Transactions. 2018;46:931–6.

6. Bernstein DB, Sulheim S, Almaas E, Segrè D. Addressing uncertainty in genome-scale metabolic model reconstruction and analysis. Genome Biology. 2021;22:1–22.

7. Pham N, van Heck RGA, van Dam JCJ, Schaap PJ, Saccenti E, Suarez-Diez M. Consistency, inconsistency, and ambiguity of metabolite names in biochemical databases used for genome-scale metabolic modelling. Metabolites. 2019;9.

8. Ravikrishnan A, Raman K. Critical assessment of genome-scale metabolic networks: The need for a unified standard. Briefings in Bioinformatics. 2015;16:1057–68.

9. Chindelevitch L, Stanley S, Hung D, Regev A, Berger B. MetaMerge: scaling up genome-scale metabolic reconstructions with application to Mycobacterium tuberculosis. Genome biology. 2012;13:1–13.

10. Sauls JT, Buescher JM. Assimilating genome-scale metabolic reconstructions with modelBorgifier. Bioinformatics. 2014;30:1036–8.

11. Mohammadi R, Zahiri J, Niroomand MJ. iMet: A graphical user interface software tool to merge metabolic networks. Heliyon. Elsevier; 2019;5:e01766.

12. Ganter M, Bernard T, Moretti S, Stelling J, Pagni M. MetaNetX.org: a website and repository for accessing, analysing and manipulating metabolic networks. Bioinformatics. 2013;29:815–6.

13. Ebrahim A, Lerman JA, Palsson BO, Hyduke DR. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. BMC Systems Biology. 2013;7:1.

14. King ZA, Dräger A, Ebrahim A, Sonnenschein N, Lewis NE, Palsson BO. Escher: A Web Application for Building, Sharing, and Embedding Data-Rich Visualizations of Biological Pathways. PLoS Computational Biology. 2015;11:1–13.

15. Cottret L, Frainay C, Chazalviel M, Cabanettes F, Gloaguen Y, Camenen E, et al. MetExplore: Collaborative edition and exploration of metabolic networks. Nucleic Acids Research. 2018;46:W495–502.

16. Luo W, Pant G, Bhavnasi YK, Blanchard SG, Brouwer C. Pathview Web: User friendly pathway visualization and data integration. Nucleic Acids Research. 2017;45:W501–8.

17. Hari A, Lobo D. Fluxer: a web application to compute, analyze and visualize genome-scale metabolic flux networks. Nucleic Acids Research. 2020;48:W427–35.

18. Seaver SMD, Liu F, Zhang Q, Jeffryes J, Edirisinghe JN, Mundy M, et al. The ModelSEED Biochemistry Database for the integration of metabolic annotations and the reconstruction, comparison and analysis of metbaolic models for plants, fungi and microbes. Nucleic Acids Research. 2020;1–14.

19. Moretti S, Martin O, Van Du Tran T, Bridge A, Morgat A, Pagni M. MetaNetX/MNXref - Reconciliation of metabolites and biochemical reactions to bring together genome-scale metabolic networks. Nucleic Acids Research. 2016;44:D523–6.

20. Moretti S, Tran VDT, Mehl F, Ibberson M, Pagni M. MetaNetX/MNXref: unified namespace for metabolites and biochemical reactions in the context of metabolic models. Nucleic acids research. 2021;49:D570–4.

21. Schellenberger J, Park JO, Conrad TM, Palsson BT. BiGG: A Biochemical Genetic and Genomic knowledgebase of large scale metabolic reconstructions. BMC Bioinformatics. 2010;11.

22. Kanehisa M, Furumichi M, Sato Y, Ishiguro-Watanabe M, Tanabe M. KEGG: integrating viruses and cellular organisms. Nucleic Acids Research. 2021;49:D545–51.

23. Hastings J, Owen G, Dekker A, Ennis M, Kale N, Muthukrishnan V, et al. ChEBI in 2016: Improved services and an expanding collection of metabolites. Nucleic Acids Research. 2016;44:D1214–9.

24. Heller SR, McNaught A, Pletnev I, Stein S, Tchekhovskoi D. InChI, the IUPAC International Chemical Identifier. Journal of Cheminformatics. Journal of Cheminformatics; 2015.

25. Hucka M, Bergmann FT, Dräger A, Hoops S, Keating SM, Le Novère N, et al. The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core. Journal of Integrative Bioinformatics. 2018;15.

26. Ebrahim A, Lerman JA, Palsson BO, Hyduke DR. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. BMC Systems Biology. BMC Systems Biology; 2013;7:1.

27. Norsigian CJ, Pusarla N, McConn JL, Yurkovich JT, Dräger A, Palsson BO, et al. BiGG Models 2020: multi-strain genome-scale models and expansion across the phylogenetic tree. Nucleic acids research. 2020;48:D402–6.

28. Duarte NC, Becker SA, Jamshidi N, Thiele I, Mo ML, Vo TD, et al. Global reconstruction of the human metabolic network based on genomic and bibliomic data. Proceedings of the National Academy of Sciences of the United States of America. 2007;104:1777–82.

29. Bordbar A, Jamshidi N, Palsson BO. iAB-RBC-283: A proteomically derived knowledge-base of erythrocyte metabolism that can be used to simulate its physiological and patho-physiological states. BMC Syst Biol. 2011;5:110.

30. Thomas A, Rahmanian S, Bordbar A, Palsson BØ, Jamshidi N. Network reconstruction of platelet metabolism identifies metabolic signature for aspirin resistance. Sci Rep. 2014;4:3925.

31. König M. fbc_curation: Reproducibility of constraint-based models. Zenodo: DOI: 10.5281/zenodo.3708271; 2022.

32. Aite M, Chevallier M, Frioux C, Trottier C, Got J, Cortés MP, et al. Traceability, reproducibility and wiki-exploration for “à-la-carte” reconstructions of genome-scale metabolic models. PLOS Computational Biology. 2018;14:e1006146.

33. Karp PD, Midford PE, Billington R, Kothari A, Krummenacker M, Latendresse M, et al. Pathway Tools version 23.0 update: software for pathway/genome informatics and systems biology. Briefings in Bioinformatics. 2021;22:109–26.

34. Machado D, Andrejev S, Tramontano M, Patil KR. Fast automated reconstruction of genome-scale metabolic models for microbial species and communities. Nucleic Acids Research. 2018;46:7542–53.

35. Wang H, Marcišauskas S, Sánchez BJ, Domenzain I, Hermansson D, Agren R, et al. RAVEN 2.0: A versatile toolbox for metabolic network reconstruction and a case study on Streptomyces coelicolor. PLOS Computational Biology. 2018;14:e1006541.

36. Hanemaaijer M, Olivier BG, Röling WFM, Bruggeman FJ, Teusink B. Model-based quantification of metabolic interactions from dynamic microbial-community data. PLOS ONE. 2017;12:e0173183.

37. Mendoza SN, Olivier BG, Molenaar D, Teusink B. A systematic assessment of current genome-scale metabolic reconstruction tools. Genome Biology. 2019;20:1–20.

38. McAllister KA, Peery RB, Zhao G. Acyl Carrier Protein Synthases from Gram-Negative, Gram-Positive, and Atypical Bacterial Species: Biochemical and Structural Properties and Physiological Implications. J Bacteriol. 2006;188:4737–48.

39. Brown S, Santa Maria JP, Walker S. Wall Teichoic Acids of Gram-Positive Bacteria. Annu Rev Microbiol. 2013;67:313–36.

40. Swoboda JG, Campbell J, Meredith TC, Walker S. Wall Teichoic Acid Function, Biosynthesis, and Inhibition. Chem Eur J of Chem Bio. 2009;11:35–45.

41. Latendresse M. Efficiently gap-filling reaction networks. BMC Bioinformatics. 2014;15:1–8.

42. Satish Kumar V, Dasika MS, Maranas CD. Optimization based automated curation of metabolic reconstructions. BMC Bioinformatics. 2007;8:1–16.

43. Zimmermann J, Kaleta C, Waschina S. Gapseq: Informed prediction of bacterial metabolic pathways and reconstruction of accurate metabolic models. Genome biology. Genome Biology; 2021;1–35.

44. Schroeder WL, Saha R. OptFill: A Tool for Infeasible Cycle-Free Gapfilling of Stoichiometric Metabolic Models. iScience. Elsevier Inc.; 2020;23:100783.

45. Kristjansdottir T, Bosma EF, Branco Dos Santos F, Özdemir E, Herrgård MJ, França L, et al. A metabolic reconstruction of Lactobacillus reuteri JCM 1112 and analysis of its potential as a cell factory. Microbial Cell Factories. BioMed Central; 2019;18:1–19.

46. Nogales J, Mueller J, Gudmundsson S, Canalejo FJ, Duque E, Monk J, et al. High-quality genome-scale metabolic modelling of Pseudomonas putida highlights its broad metabolic capabilities. Environmental Microbiology. 2020;22:255–69.

47. Nogales J, Palsson BØ, Thiele I. A genome-scale metabolic reconstruction of Pseudomonas putida KT2440: iJN746 as a cell factory. BMC Syst Biol. 2008;2:79.

48. Payne DD, Renz A, Dunphy LJ, Lewis T, Dräger A, Papin JA. An updated genome-scale metabolic network reconstruction of Pseudomonas aeruginosa PA14 to characterize mucin-driven shifts in bacterial metabolism. NPJ Syst Biol Appl. 2021;7:37.

49. Chung WY, Zhu Y, Mahamad Maifiah MH, Shivashekaregowda NKH, Wong EH, Abdul Rahim N. Novel antimicrobial development using genome-scale metabolic model of Gram-negative pathogens: a review. Journal of Antibiotics. 2021;74:95–104.

50. Norsigian CJ, Kavvas E, Seif Y, Palsson BO, Monk JM. iCN718, an Updated and Improved Genome-Scale Metabolic Network Reconstruction of Acinetobacter baumannii AYE. Front Genet. 2018;9:121.

51. Wilken StE, Monk JM, Leggieri PA, Lawson CE, Lankiewicz TS, Seppälä S, et al. Experimentally Validated Reconstruction and Analysis of a Genome-Scale Metabolic Model of an Anaerobic Neocallimastigomycota Fungus. mSystems. 2021;6:1–22.

52. Derrer B, Macheroux P, Kappes B. The shikimate pathway in apicomplexan parasites: implications for drug development. Front Biosci (Landmark Ed). 2013;18:944–69.

53. Serer MI, Carrica M del C, Trappe J, López Romero S, Bonomi HR, Klinke S, et al. A high-throughput screening for inhibitors of riboflavin synthase identifies novel antimicrobial compounds to treat brucellosis. The FEBS Journal. 2019;286:2522–35.

54. Bernstein DB, Sulheim S, Almaas E, Segrè D. Addressing uncertainty in genome-scale metabolic model reconstruction and analysis. Genome Biology. Genome Biology; 2021;22:1–22.

55. Hwang J, Hari A, Cheng R, Gardner JG, Lobo D. Kinetic modeling of microbial growth, enzyme activity, and gene deletions: an integrated model of β-glucosidase function in Cellvibrio japonicus. Biotechnology and Bioengineering. 2020;bit.27544.

56. Ko JM, Mousavi R, Lobo D. Computational Systems Biology of Morphogenesis. In: Cortassa S, Aon MA, editors. Computational Systems Biology in Medicine and Biotechnology: Methods and Protocols. New York, NY: Springer US; 2022. p. 343–65.

57. Ko JM, Lobo D. Continuous Dynamic Modeling of Regulated Cell Adhesion: Sorting, Intercalation, and Involution. Biophysical Journal. Biophysical Society; 2019;117:2166–79.

58. Mousavi R, Konuru SH, Lobo D. Inference of dynamic spatial GRN models with multi-GPU evolutionary computation. Briefings in Bioinformatics. 2021;22:1–11.

59. Bostock M, Ogievetsky V, Heer J. D3 data-driven documents. IEEE Transactions on Visualization and Computer Graphics. IEEE; 2011;17:2301–9.