۱- توضیح کد ها :

سه کد به زبان جولیا و یک کد به زبان پایتون نوشته شده است. کد های جولیا بخش ها و تابع ها مشترکی دارند پس اول کارکرد بخش و تابع های مشترک را توضیح میدهیم و سپس وارد جزییات هر کد میشویم.

بخش های مشترک کد های جولیا:

استراکت models:

این استراکت شامل فیلدهای S, Reactions, Metabolites, Genes, m, n, ub, lb است که در هنگام لود یک مدل یک شی از این استراکت ساخته میشود و فیلد های مورد نظر شی با توجه به مدل مقداردهی میشود.

تابع loadmyModel:

این تابع یک رشته name که نام مدل مورد نظر است را از ورودی دریافت میکند و درصورتی که قبلا دانلود نشده باشد آن را دانلود میکند و به فرمت استاندارد COBREXA لود میکند و سپس یک شی از استراکت models به نام myModel ساخته و فیلد های آن را با توجه به مدل مقداردهی کرده و در خروجی نهایی آبجکت myModels را که شامل فیلدهای ذکر شده است برمیگرداند.

تابع if_rev :

هدف این تابع بررسی برگشت پذیر بودن یا نبودن واکنش هاست. در ورودی یک مدل گه شی خروجی تابع قبل است را دریافت میکند سپس lb تمام واکنش ها را بررسی میکند. اگر lb<0 یعنی واکنش برگشت پذیر است پس درایه ی نظیر این واکنش را در آرایه ی rev را یک قرار میدهیم و در غیر صورت درایه صفر مقداردهی خواهد شد. در نهایت آرایه ی rev در خروجی داده میشود که با توجه به درایه های آن میتوان برگشت پذیر بودن یا نبودن واکنش ها را بررسی کرد.

تابع hemogen :

هدف این تابع همگن کردن مدل است. در ورودی یک مدل، مقدار M و آرایه ی rev که با تابع قبلی ساخته شده و نشان دهنده ی برگشت پذیر بودن یا نبودن واکنش هاست را دریافت میکند. با یک حلقه روی rev بررسی

میکند هر واکنش برگشت پذیر هست یا نه. اگر برگشت پذیر باشد lb=-M , ub=M و اگر برگشت پذیر نباشد ub=M , lb=0 در مدل ست میشود. در نهایت مدل همگن شده در خروجی داده میشود.

کد printModels.jl :

هدف این کد صرفا دانلود و چاپ مدل ها در خروجی است. ئر یک حلقه با تابع loadmyModels تمام مدل های خواسته شد دانلود و چاپ میشوند. خروجی این بخش به صورت فایل models.txt ضمیمه شده است.

کد classic.jl :

هدف این کد پیاده سازی مدل کلاسیک و یافتن واکنش های بلاک شده از این طریق است. در این کد ابتدا مدلی که میخواهیم را با تابع loadmyModel لود میکنیم، سپس با تابع if_rev واکنش های برگشت پذیر و ناپذیر آن را مشخص کرده و با تابع Hemogen آن را همگن میکنیم. سپس با تابع irreversable_classic واکنش های برگشت ناپذیر بلاک شده و با reversable_classic واکنش های برگشت پذیر بلاک شده را به روش کلاسیک پیدا کرده و چاپ میکنیم که خروجی های روش کلاسیک در فایلی به نام cls.txt ضمیمه شده و در آن خروجی ها به تفکیک مدل و برگشت پذیر یا ناپذیر بودن واکنش ها مشخص است. در ادامه دو تابع پیاده شده در این کد توضیح داده میشوند :

تابع irreversable_classic :

ورودی این تابع یک مدل و آرایه ی rev نظیر آن است که مشخص میکند هر واکنش برگشت پذیر هست یا نه. روی تمام واکنش ها یک حلقه داریم و اول بررسی میکنیم واکنش برگشت پذیر هست یا نه. اگر برگشت ناپذیر بود یک مدل برنامه ریزی خطی به نام modelLP با solver GLPK برای آن تعریف میکنیم. سپس متغیر های مدل را تعریف میکنیم که آرایه ی V به طول واکنش هاست. سپس تابع هدف تعریف میشود که بیشینه کردن V[i] است (i اندیس واکنشی است که مدل برای آن تشکیل دادیم). پس از آن هم قید ها می آیند که یک قید mass balance داریم S.V=0 و V[i]<=1 و اینکه تمام درایه های V بین lb و ub تعریف شده ی خود باشند. سپس مدل را حل میکنیم و چک میکنیم اگر V[i] به دست آمده برای آن واکنش با یک tolerance مشخصی برابر با ۰ بود، آن را به عنوان واکنش بلاک شده چاپ میکنیم.

تابع reversable_classic :

ورودی این تابع یک مدل و آرایه ی rev نظیر آن. یک حلقه داریم و اول بررسی میکنیم واکنش برگشت پذیر هست یا نه. اگر برگشت پذیر بود دو مدل باید برای آن حل کنیم. یکی در جهت رفت و دیگری در جهت برگشت.

مدل در جهت رفت که عینا مانند قسمت قبل است. برای جهت برگشت نیز تنها تابع هدف و یکی از قیود تغییر میکند. قید کمینه کردن V[i] خواهد بود و قید V[i]<=1 به V[i]>=-1 تبدیل شده است.در نهایت بررسی میکنیم اگر V[i] واکنش هم در جهت رفت و هم در جهت برگشت با tolerance مشخصی برابر با صفر بود آن واکنش به عنوان واکنش بلاک شده معرفی میشود. (در کد اینگونه پیاده سازی شده که مجموع V[i] رفت و برگشت چک شده با تلرانس مورد نظر صفر هست یا نه)

بدیهی است که مدل های با استفاده از JuMP و اپتیمایزر GLPK نوشته و حل شده اند.

کد modern.jl :

این کد مدلی که میخواهیم را با loadmyModel لود میکند و با استفاده از تابع if_rev واکنش های برگشت پذیر و ناپذیر آن را مشخص کرده و با تابع Hemogen آن را همگن میکنیم. در نهایت با استفاده از تابع irreversable_modern با استفاده از روش مدرن واکنش های برگشت ناپذیر بلوکه شده ی آن را پیدا و چاپ میکنیم. خروجی های روش مدرن در فایلی به نام modern.txt ضمیمه شده و در آن خروجی ها به تفکیک مدل مشخص است. در ادامه دو تابع پیاده شده در این کد توضیح داده میشوند :

تابع irreversable_modern:

این تابع به عنوان ورودی یک مدل و آرایه ی rev نظیر آن را میگیرد که مشخص باشد کدام واکنش ها برگشت پذیر هستند و کدام واکنش ها نه. تعداد واکنش های برگشت ناپذیر شمرده برای استفاده های بعدی شمرده میشود. آرایه irr یک آرایه ی کمکی است به طول واکنش های برگشت ناپذیر که در آن اندیس واکنش های برگشت ناپذیر ذخیره میشوند و بعد در تعریف قید ها استفاده میشود. I نیز یک بردار به طول واکنش های برگشت ناپذیر است که با ۱ پر شده و برای تعریف تابع هذف استفاده میشود. یک مدل برنامه ریزی خطی به نام modelLP با solver GLPK تعریف میشود که در آن دو متغیر V و u را داریم. تابع هدف بیشینه کردن $I^T.u$ است که درواقع بیشینه کردن مجموع درایه های u است. یک قید mass balance داریم S.V=0 و سایر قید ها یکی این است که تمام درایه های u بین ۰ و ۱ باشند و دیگری این است که به ازای تمام واکنش های برگشت ناپذیر v[i]>u[i] باشد. سپس مدل را حل میکنیم. و بررسی میکنیم اگر V[i] برای واکنش برگشت پذیری با تلرانس مشخصی صفر بود، آن واکنش را به عنوان بلوکه شده در خروجی چاپ میکنیم.

نکته: در تمام رویکرد های بالا tolerance و M برای هرکدام از شبکه های متابولیکی به طور دستی با ازمون و خطا ست شده است که در بخش نتایج به ان میپردازیم.

کد final_check.py:

این کد به زبان پایتون نوشته شده است و هدف از آن بررسی و مقایسه ی نتایج به دست آمده از روش های قبلی است. در این کد ابتدا خروجی های نتیجه شده از cobra به ازای تمامی مدل ها محاسبه و در فایل cobra.txtذخیره میشود. سپس فایل های خروجی قسمت های قبل (modern.txt و cls.txt) خوانده شده و تعداد واکنش های پیدا شده به تفکیک مدل و برگشت پذیر بودن یا نبودن آن ها محاسبه شده و در مقایسه با یکدیگر در خروجی چاپ میشوند. خروجی این بخش و نتایج حاصل از آن در ادامه توضیح داده شده است.

۲-خروجی و نتایج :

همانطور که در بخش قبل ذکر شد، خروجی تمام متد ها در فایل های مربوطه ذخیره شده است.

الف) واکنش های پیدا شده

تعداد واکنش های بلوکه یافته شده بین حالت کلاسیک و cobaPy و هم چنین برگشت ناپذیرهای کلاسیک و مدرن برای مقایسه در جدول زیر آورده شده است.

	e_coli_core	iAB_RBC_283	iNF517	iNJ661	Recon3D
cobraPY(all)	8	16	241	285	0
Classic_all	8	23	246	292	Time_limit
Classic_irreversable	8	2	162	207	Time_limit
Modern(irreversible)	8	2	166	234	0

میتوان دید نتیجه ی cobraPy و مدل کلاسیک کاملا یکسان یا با اختلاف کمی یکسان است. هم چنین نتیجه ی روش مدرن و واکنش های برگشت ناپذیر یافته شده در روش مدرن نیز یکسان یا با اخلاف کمی یکسان هستند.

برای مثال برای حالت برگشت ناپذیر در iAB_RBC_283 هر دو مدل کلاسیک و مدرن واکنش های AP4AH1 و ICDHyr را در خروجی برگرداندند.

ب) زمان اجرا

زمان اجرای روش مدرن از روش کلاسیک بسیار بیشتر است که این منطقی است چون تنها یک مدل را حل میکند اما مدل کلاسیک به تعداد واکنش های برگشت ناپذیر+دوبرابر واکنش های برگشت پذیر باید مدل حل کند. میتوان دید در مورد Recon3D مدل کلاسیک پس از تایم لیمیت ست شده ۵ ساعته به جواب نهایی نرسیده اما مدل مدرن در حدود ۴۰ دقیقه پاسخ نهایی را بازگرداند.

ج) مقدار M و tolerance در آزمایش ها

	e_coli_core	iAB_RBC_283	iNF517	iNJ661	Recon3D
Classic	M=10^6 Tolerance=0	M=10^2 Tolerance=0	M=10^6 Tolerance=1e-10	M=10^6 Tolerance=1e-12	Time_limit
Modern	M=10^6 Tolerance=0	M=10^6 Tolerance=0	M=10^6 Tolerance=0	M=10^6 Tolerance=0	M=10^6 Tolerance=0