حمید رضا کاشانی 810100441

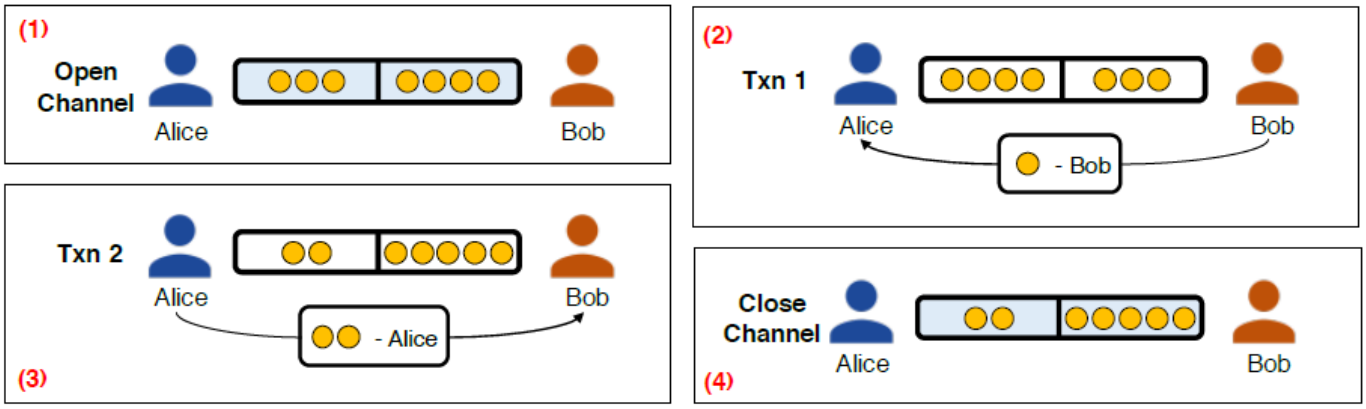
پروژه دوم

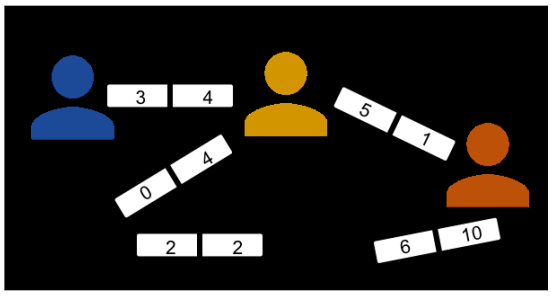
توضیحات ابتدایی برای اشنایی با موضوع پروژه

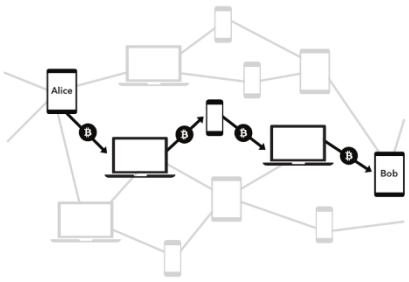
شبکه های کانال پرداخت یا PCN ها شبکه ای از نود هایی هستند که دو به دو با یکدیگر کانال پرداخت ایجاد می کنن و بدون تکرانش در زنجیره بلاک قابلیت پرداخت را در شرایطی داخل نود های شبکه می دهد. این شبکه به علت مشکل مقیاس پذیری در برخی رمز ارز ها مانند بیتکون ایجاد شده است زیرا هر بلاک درزمانی با میانگین ثابت و حجم محدود رمز ارز ایجاد می شود و با زیاد شدن تقاضا بلاکچین نمی تاند اسکیل پیدا کنند و باید مشترکین برای پرداخت در صف قرار بگیرن .

این شبکه به این صورت کار می کند که هر نود با چند نود دیگر در شبکه کانال ایجاد کرده است و در این شبکه هر دو نودی که بخواهند با هم تراکنش داشته باشند باید ارز های خود را با انتقال ار بین کانال های موجود به نود مورد نظر برساند و هر نود میانی هزینه انتقال ارز از خود را که قبلا اعلام کرده است از ارز های انتقالی بر می دارد و به نود بعدی می دهد .

نحوه ی ایجاد کانال اینگونه است که هر دو نود مقداری ارز را در حسابی چند امضا در زنجیره بلاک چین قرار می دهند برای مثال هر نود ده ارز را در حساب کانال قرار می دهد و یک کانال با ظرفیت 20 را تشکیل می دهد که هر نود می تواند ده ارز را هنگام تراز بودن کانال از خود انتقال دهد و به طور مثال با عبور دو ارز از یک طرف تعادل کانال به هم ریخته و به 8 و 12 تبدیل می شود و در صورتی که یک طرف کانال 0 شود دیگر نمیتوان از آن طرف کانال ارز انتقال داد مگر این که کانال جدید بین دو نود در بلاکچین ایجاد شود و کانال قبلی بسته شود که این عمل برای دو نود هزیته ایجاد می کند .







پس مسیریابی ها در شبکه باید به گونه ای باشد که تعادل کانال ها حفظ شود که ابن به این معنی است که مقدار ارز عبوری از کانال از دو نود برابر شود . همینطور این را هم باید در نظر گرفت که هر کانال محدودیتی دارد در انتقال ارز از خود که متناسب با ظرفیت آن است.

مدل کردن مسئله:

شبکه خود را به صورت گراف جهت دار وزن دار در نظر میگریم که نود های آن افراد در شبکه لایتنیگ هستن و هر یال یک طرف از کانال موجود بین دو نفر را نشان می دهد. وزن هر یال مقدار ارزی است که می شود از هر نود به نوددیگر انتقال داد و توجه شود شود که به علت وجود بالانس در موجودی هر کانال با کم شدن مقدار وزن یا ظرفیت هر یال به جهت مخالف آن همان مقدار اضافه می شود.مسیر انتخاب شده بین دو نود را مجموعه ای از یال های سری شده تشکیل می دهد. هر نود می خواهد مقداری ارز را به نود دیگری بدهد که به آن تقاضا می گوییم و با dij نمایش می دهیم. در شبکه از نود به نود دیگر مسیر هایی وجود دارد که هر کدام از این مسیر ها مقداری فلو ارز در آن ها وجود دارد که این مقدار را با Xp نمایش می دهیم. در هر نود برای جابه جایی ارز مقداری طول می کشد تا این تراکنش انجام شود و این پول باید تا زمان برگشت رسید قفل بماند که میانگین آن را با دلتا نمایش می دهیم و این دلتا باعث کم شدن ظرفیت واقعی کانال ها می شود به صورت میانگین. مقدار کل ظرفیت یک کانال یعنی مجموع ارزی که دو نود در کانال گذاشته اند برای تشکیل کانال را ظرفیت کانال می گویند و با Cuv نمایش داده شده است.xuv هم مقدار مجموع فلویی است که از نود u به نودv وجود دارد دقت شود که نود vوu متصل اند.

جدول توضیحات هر پارامتر

$G(V, E)$	Graph of the PCN with a set of V routers and E payment channels
$\mathcal{P}_{ij}$	Set of paths that sender i uses to receiver j
$\mathcal{P}$	$\bigcup_{i,j \in V} \mathcal{P}_{ij}$
x_p	Average rate of transaction-units on path p between sender i and receiver j
x_{uv}	$\sum_{p \in \mathcal{P}:(u,v) \in p} x_p$
d_{ij}	Demand from sender i to receiver j
c_{uv}	Total amount of tokens escrowed into payment channel (denotes channel size)(u, v)
Δ	Average time (s) over which tokens sent across a payment channel are unusable

Table 4.1: Notation for routing problem

حمید رضا کاشانی **پروژه دوم** 810100441

هدف ما این است که مجموع کل فلو های عبوری از شبکه را به ماکسیموم برسانیم (به صورت منصافانه یا غیر منصافنه که آن را با مشخص کردن تابع U می شود مدل کرد) و قید هایی در مسئله وجود دارد که باید این قید ها هم در شبکه اعمال شود:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i,j \in V} U\Big(\sum_{p \in \mathcal{P}_{ij}} x_p\Big) \\
\text{s.t.} \quad & \sum_{p \in \mathcal{P}_{ij}} x_p \le d_{ij} \quad \forall i,j \in V \\
& x_{uv} + x_{vu} \le \frac{c_{uv}}{\Delta} \quad \forall (u,v) \in E \\
& x_{uv} = x_{vu} \quad \forall (u,v) \in E \\
& x_p \ge 0 \quad \forall p \in \mathcal{P}.
\end{aligned}$$

قید اول برای این است که مجموع فلو های دو نفر بیشتر از تقاضایشان نشود.

قید دوم محدودیت ظرفیت هر کانال را مشخص می کند که مجموع فلو عبور از هر دو طرف کانال باید از ظرفیت حقیقی آن کانال که وابسته به میانگین زمان تراکنش است کوچکتر باشد.

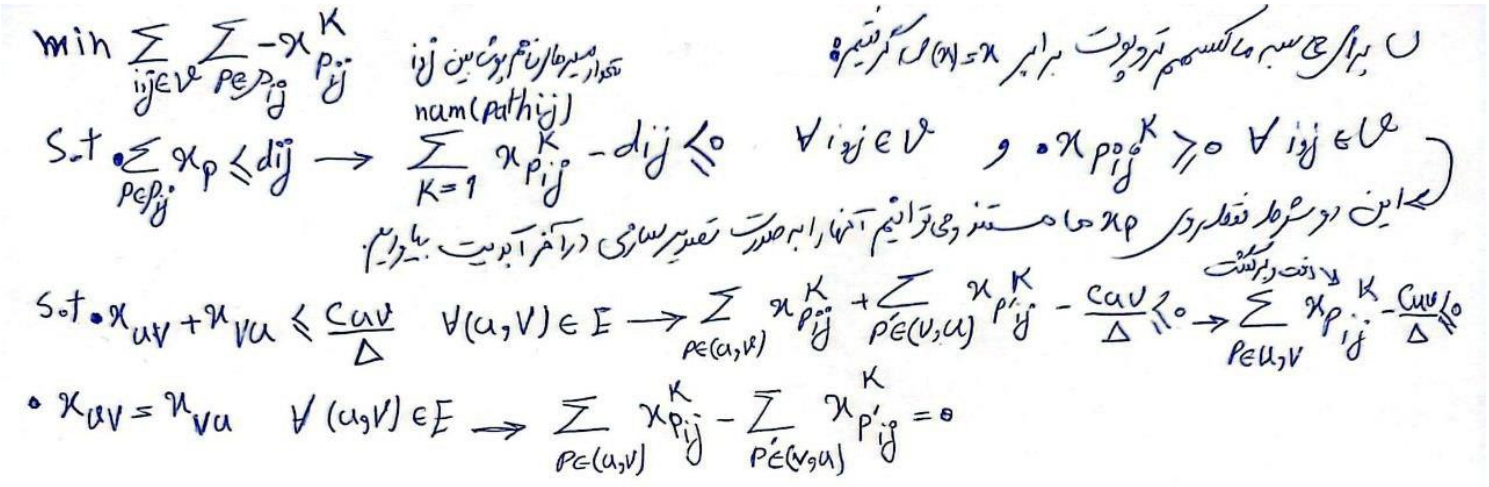
قید سوم برای پایداری بیشتر شبکه در نظر گرفته شده است که مقدار فلو عبور از هر دو طرف کانال با هم برابر باشد.

قید چهارم مقدار فلو ها نمی تواند منفی شود.

توجه شود که این چهار قید برای هر دو نود باید برقرار باشد

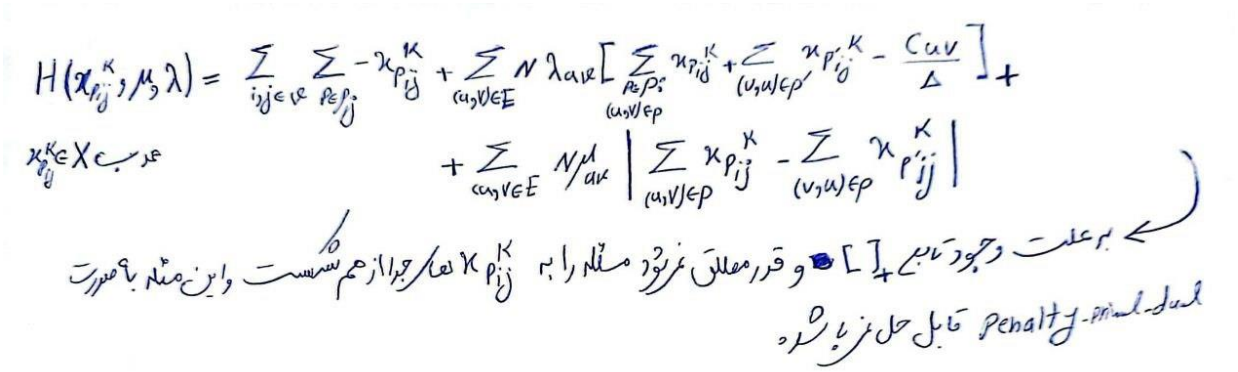
حل مسئله:

صورت مسئله



حمید رضا کاشانی **پروژه دوم** 810100441

حل penalty-primal-dual به این علت نا ممکن است:



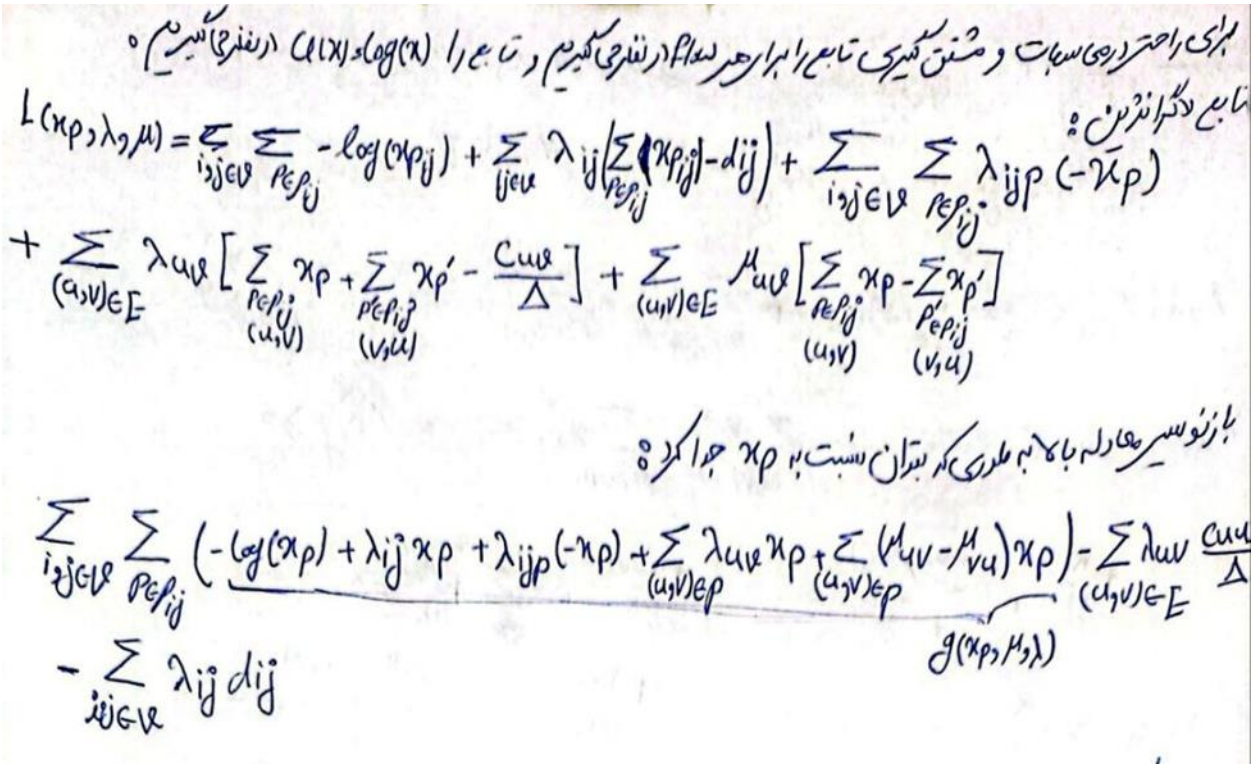
همان طور که مشاهده کردید در صورت استفاده از الگوریتم پنالتی-پریمال-دوال دیگر نمی شود مسئله را به فلو های عبوری تجزیه کرد .

حل جایگزین:

ما متغییر های دوال مسئله را به صورت توزیع شده آپدیت میکنیم و برای آپدیت بعدی مقدار مشتق را به صورت بسته به دست آورده ایم و جایگذاری می کنیم تا سیستم به جواب بهینه همگرا شود.

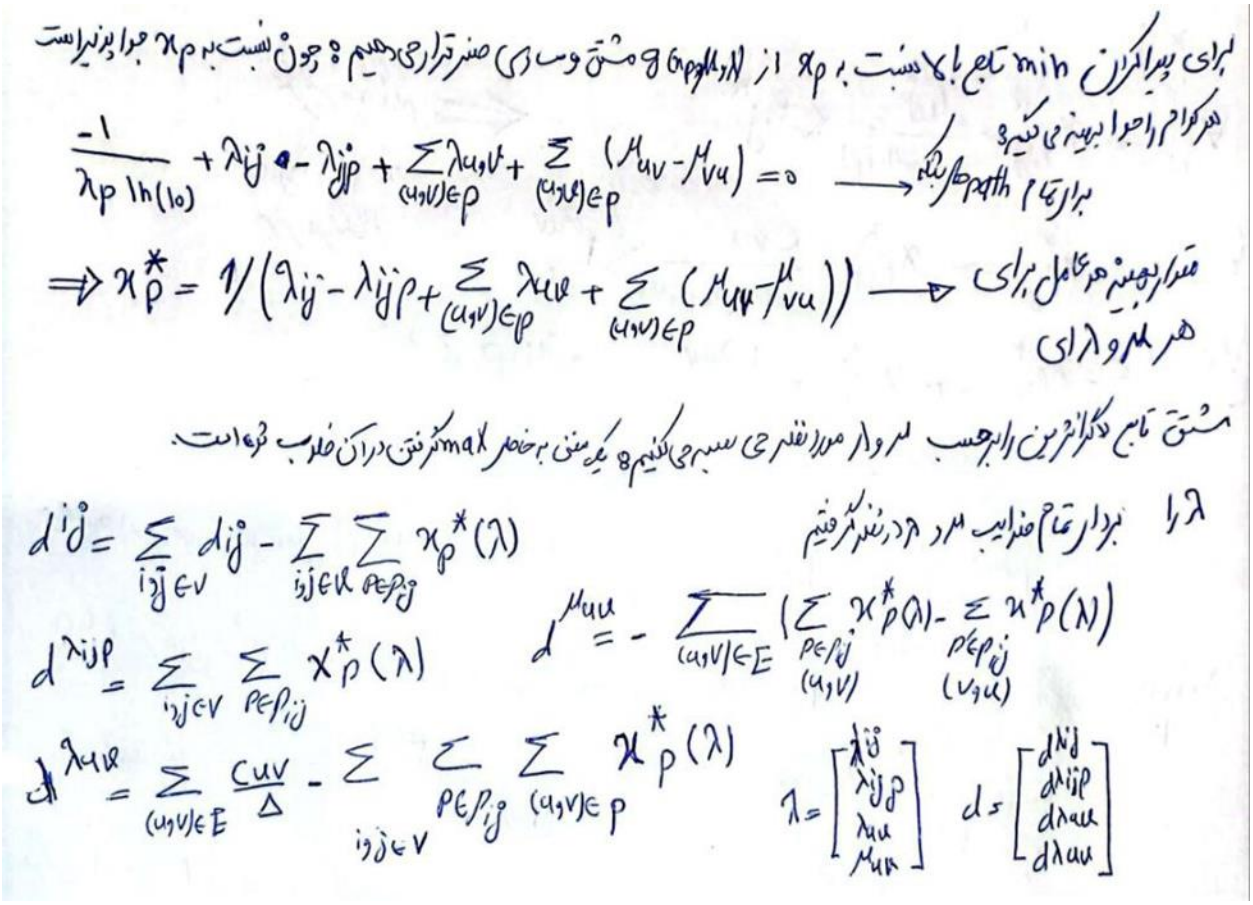
همانند پروژه اول تابع هزینه را به log تغییر می دهیم تا بتوانیم به صورت بسته مشتق را محاسبه کنیم ،تابع هزینه log فلو ها را به صورت منصفانه بهینه می کند.

و همینطور ما در این مسئله دو شرط مربوط به متغییر پریمال خود را هم باید دخیل کنیم.

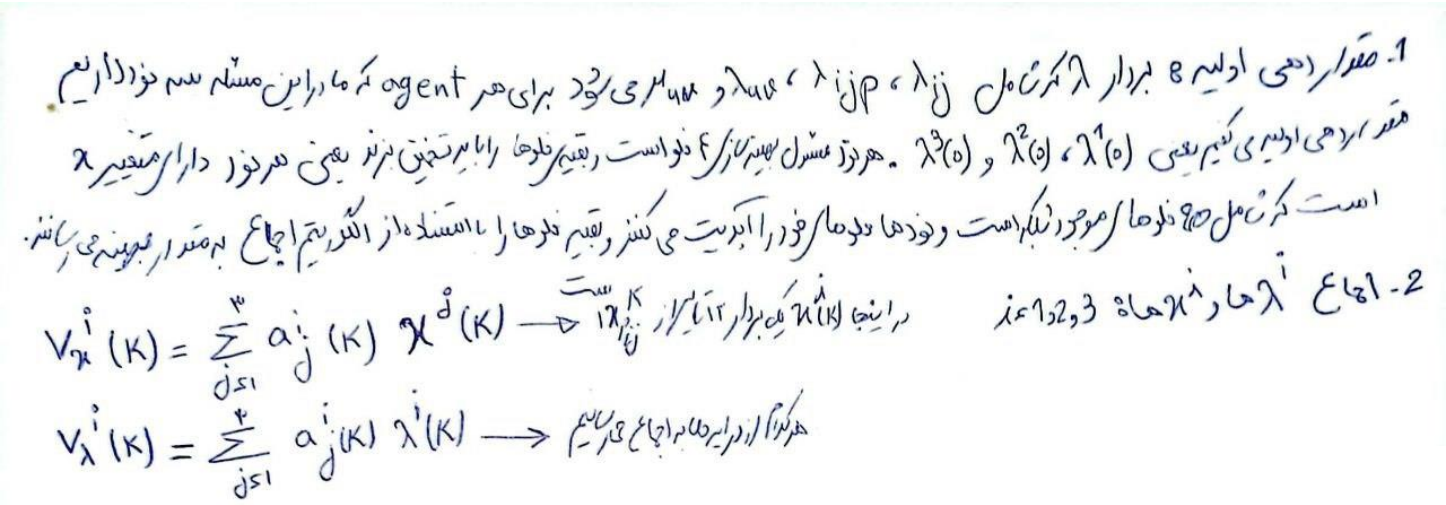


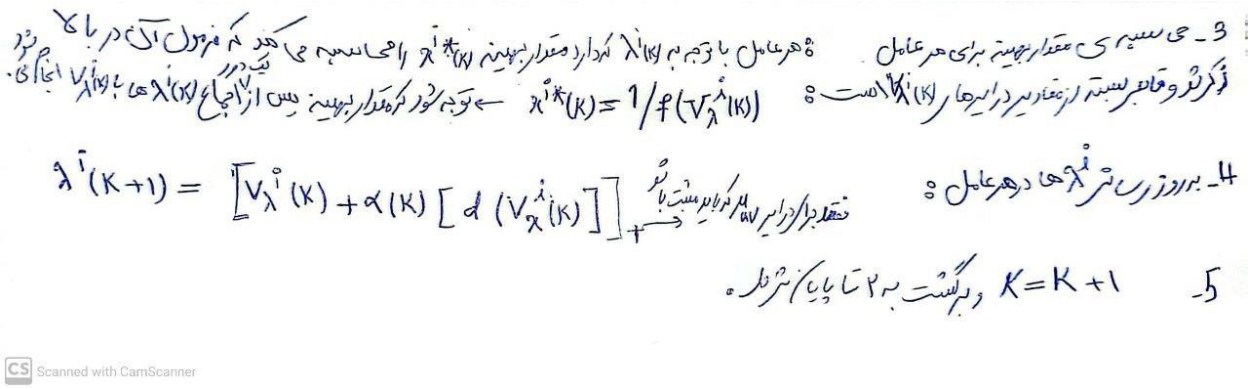
حمید رضا کاشانی 810100441

پروژه دوم



حال باید الگوریتم توزیع شده را اجرا کنیم که برای راحتی کار ما سه نود و 12 فلو را در نظر می گیریم:





که این شبیه سازی انجام شده است.

حل dual-primal:

تابع لاگرانژ را با حذف قیود مربوط به فلو ها و مثبت کردن ضریب قید تعادل با اضافه کردن دو قید تا تساوی به جای یک قید تساوی به صورت زیر تعریف می کنیم.

برای محاسبه نقطه زینی تابع لاگرانژ باید نسبت به x محدب باشد که این یعنی باید ضرایب قیود منفی نباشند برای همین شرط مساوی را به دو شرط نا مساوی تبدیل می کنیم تا بتوانیم از این روش نقطه زینی را پیدا کنیم.

صورت بندی مسئله به این شکل است :

$U(x) = x$. Consider the partial Lagrangian of the LP:

$$
\begin{aligned}
L(\mathbf{x}, \lambda, \mu) = & \sum_{i,j \in V} \sum_{p \in \mathcal{P}_{i,j}} x_p \\
& - \sum_{(u,v) \in E} \lambda_{uv} \left[\sum_{\substack{p \in \mathcal{P}: \\ (u,v) \in p}} x_p + \sum_{\substack{p' \in \mathcal{P}: \\ (v,u) \in p'}} x_{p'} - \frac{c_{u,v}}{\Delta} \right] \\
& - \sum_{(u,v) \in E} \mu_{uv} \left[\sum_{\substack{p \in \mathcal{P}: \\ (u,v) \in p}} x_p - \sum_{\substack{p' \in \mathcal{P}: \\ (v,u) \in p'}} x_{p'} \right] \\
& - \sum_{(u,v) \in E} \mu_{vu} \left[\sum_{\substack{p \in \mathcal{P}: \\ (v,u) \in p}} x_p - \sum_{\substack{p' \in \mathcal{P}: \\ (u,v) \in p'}} x_{p'} \right],
\end{aligned}
$$

ضریب مربوط به ظرفیت لینک ها به جهت کانال مرتبط نیست ولی ضریب تعادل متناسب با جهت کانال متفاوت است.

معادله بالا را به صورت زیر بازنویسی می کنیم:

$$L(\mathbf{x},\lambda,\mu)=\sum_{i,j\in V}\sum_{p\in\mathcal{P}_{i,j}}x_p\left(1-\sum_{(u,v)\in p}\lambda_{uv}-\sum_{(u,v)\in p}\mu_{uv}+\sum_{(v,u)\in p}\mu_{vu}\right)+\sum_{(u,v)\in E}\lambda_{uv}\frac{c_{uv}}{\Delta}.$$

تا بتوان آن را به فلو های عبوری تجزیه کنیم و با توجه به تغییر معادله بالا ضریب زیر را تعریف می کنیم:

$$z_p = \sum_{(u,v):(u,v)\in p}\left(\lambda_{uv}+\mu_{uv}-\mu_{(v,u)}\right)$$

ابدیت پرایمال:

$$x_p(t+1)=x_p(t)+\alpha(1-z_p(t)) \tag{9.3}$$

$$x_p(t+1)=\mathrm{Proj}_{\chi_{i,j}}(x_p(t+1)), \tag{9.4}$$

where Proj is a projection operation on to the convex set $\{x_p : \sum_{p:p\in\mathcal{P}_{i,j}} x_p \le d_{i,j}, x_p \ge 0 \ \forall p\}$, to ensure the rates are feasible.

نود های ابتدایی و انتهایی مسیر مقدار فلو فرستاده خود را متناسب با شرایط تغییر کرده در شبکه اپدیت می شنود.

اپدیت دوال:

مقدار تخمین هر نود میانی از شرایط کانال به صورت زیر انجام می شود

$$w_{uv}(t)=\sum_{\substack{p\in\mathcal{P}:\\(u,v)\in p}}x_p(t)+\sum_{\substack{p'\in\mathcal{P}:\\(v,u)\in p'}}x_{p'}(t)-\frac{c_{uv}}{\Delta}$$

$$y_{uv}(t)=\sum_{\substack{p\in\mathcal{P}:\\(u,v)\in p}}x_p(t)-\sum_{\substack{p'\in\mathcal{P}:\\(v,u)\in p'}}x_{p'}(t)$$

نود های میانی پارامتز های دوال شبکه را آپدیت می کنند تا شرایط شبکه را به سمت هدف خود تغییر دهند و ایم تغییر در اپدیت فلو نود های ابتدایی ظاهر شود:

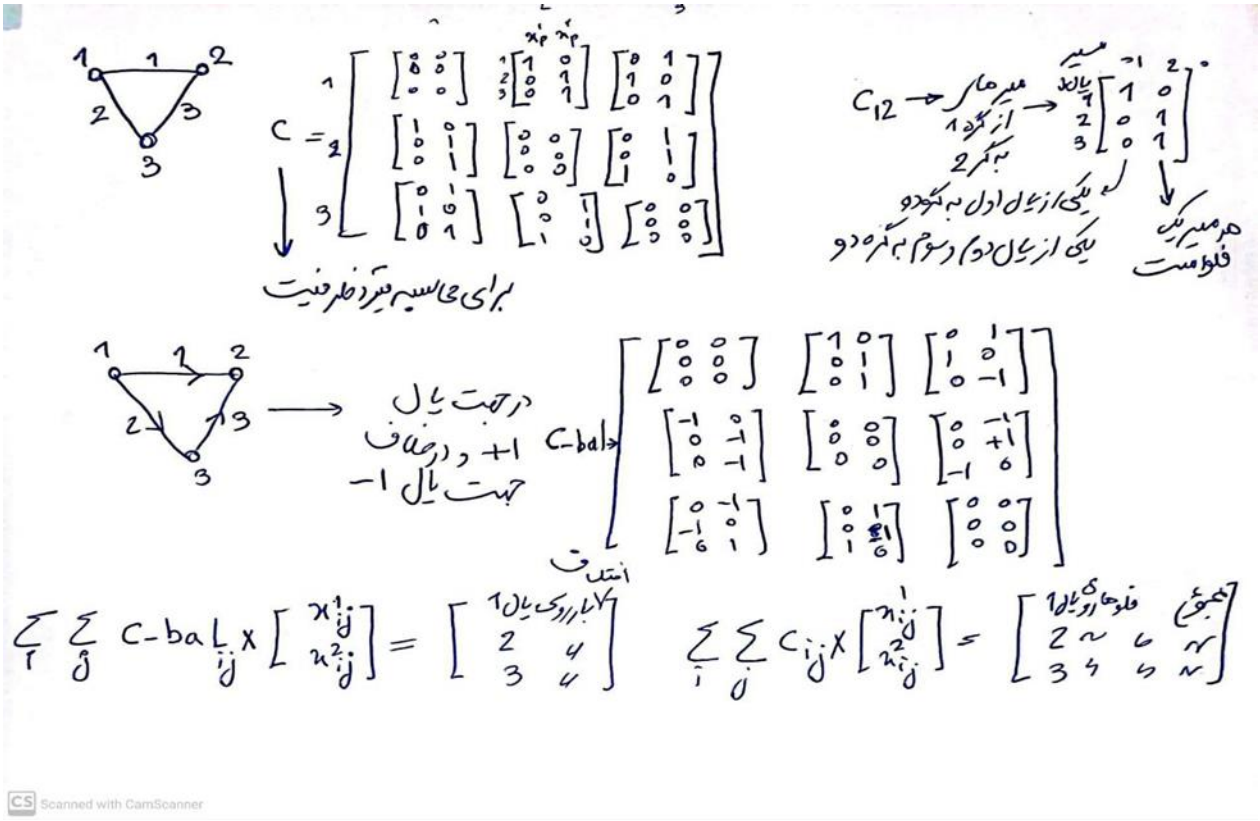
حمید رضا کاشانی **پروژه دوم** 810100441

$$\lambda_{uv}(t+1) = [\lambda_{uv}(t) + \eta w_{uv}(t)]_+$$
$$\mu_{uv}(t+1) = [\mu_{uv}(t) + \kappa y_{uv}(t)]_+$$
$$\mu_{vu}(t+1) = [\mu_{vu}(t) - \kappa y_{uv}(t)]_+ .$$

این روش سیستم ما رو به سمت نقطه زینی تابع لاگرانژ حرکت می دهد و برای استفاده عملی کارا تر است .

پیاده سازی برای سه نود و 12 فلو:

پیاده سازی الگوریتم توریع شده این مسئله به صورت دوال:



هر نود یه تخمینی ابتدایی از مقدار بردار x که شامل مقدار فلو عبوری از همه مسیر هاست دارد.

ما برای نمایش هر مسیر هر فلو به یک ستون سه درایه ای نیاز داریم و از هر تود به نود دیگر هم دو مسیر موجود دارد به جز از هر نود به خودش پس یک ماتریس مسیر c درست می کنیم که می گوید مسیر از هر نود به نود دیگر چگونه است.

حمید رضا کاشانی پروژه دوم 810100441

برای قید ظرفیت نیازی به مشخص کردن جهت مسیر ها نبود و فقط حمع هر دو طرف نیاز بود پی یک را برای نشان دادن عبور فلو از آن یال می گذاریم و صفر را برای نگذشتن آن اما برای قید تعادل چون اختلاف دو طرف مهم است باید جهت مخص شود فلو های در جهت یال مثبت و فلو های خلاف جهت یال منفی.

هر نود تخمین خود را دارد و در هر مرحله یک اجماع روی کل درایه ها انجام می دهد و یک آپدیت روی درایه های مربوط به نود خود زیرا فقظ مسیر فلو های خود را دارد .

ماتریس A را سه در سه تعریف می کنیم که هر درایه آن می گید که چه ضریبی را از همسایه خود دراجماع بگیرد.

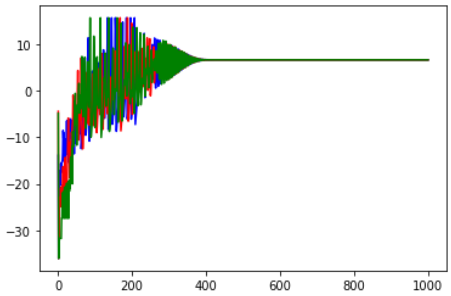
ماتریس تصادفی دوبل و متصل و تا متغییر با زمان :

```
A = np.array([[1/3,1/3,1/3],[1/3,1/3,1/3],[1/3,1/3,1/3]])
```

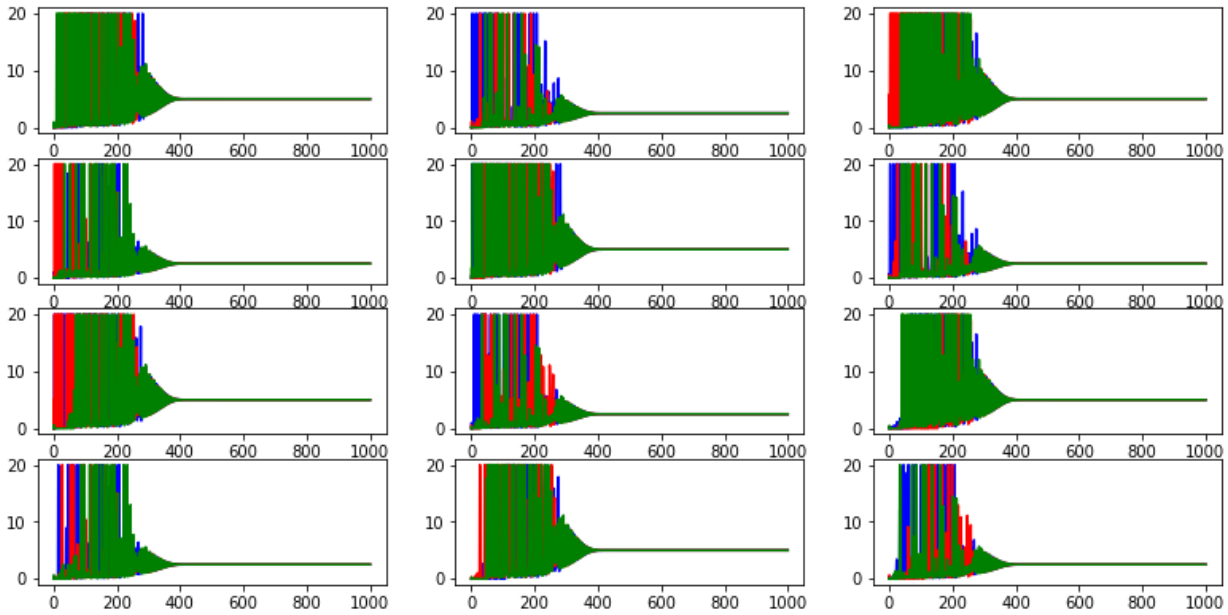
این شبیه سازی را برای 1000 ایتریشن انجام دادیم و حاصل موفقیت آمیز بود:

هر رنگ در هر نمودار نشان دهند مقدار هر نود برای آن نمودار است :

مقدار تابع ما به این صورت همگرا شدن برای هر سه نود در یک نمودار:

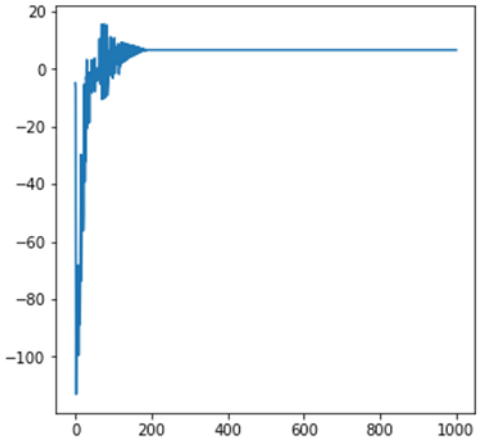


مقدار هر یک از 12 فلو برای هر نود به صورت زیر همگرا شده است:

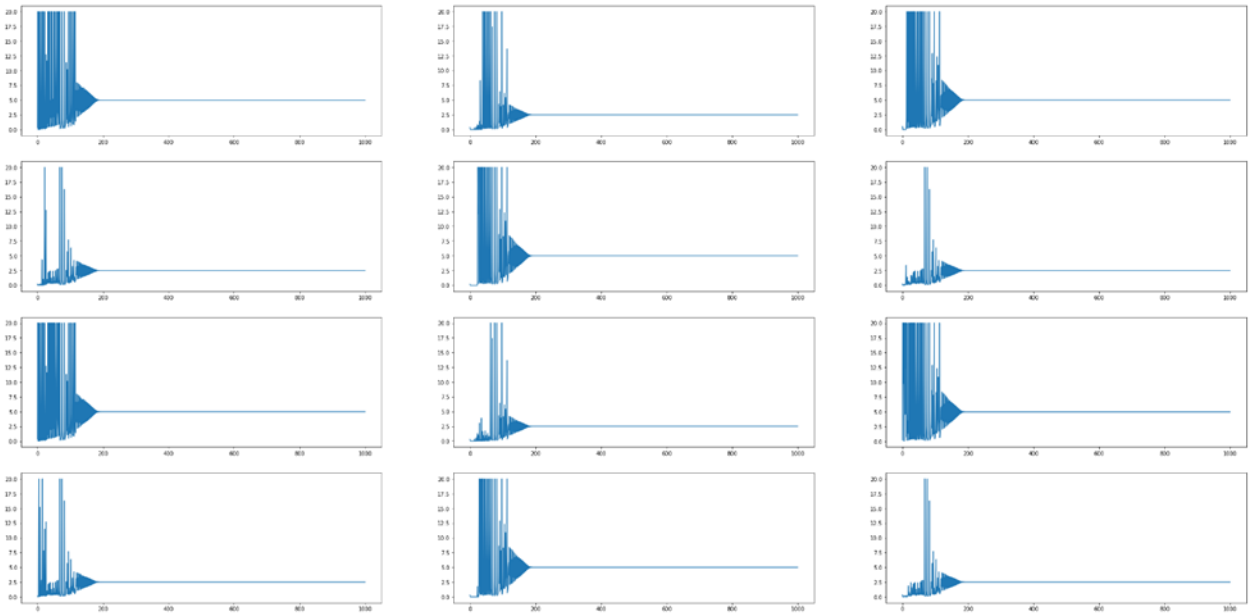


که دقیقا همان نتیجه ای است که از حل مرکزی به دست آمده است در پروژه قبلی :

مقدار نمودار تابع بهینه سازی برای حل مرکزی:



و مقدار هر یک از فلو ها پس از همگرایی با حل مرکزی:



همانطور که مشاهده می کنید مقادیر به دست آمده یکسان است که این نشان میدهد الگوریتم رفته شده برای حل توریع شده درست بود است.

و مشاهده می شود که زمان همگرایی برای الگوریتم حل مرکزی نصف زمان هگرایی الگوریتم توزیع شده است که این نشان میدهد الگوریتم توریع شده کند تر بوده و برای کار برد های سریع ضعیف تراست .

حمید رضا کاشانی 810100441

پروژه دوم

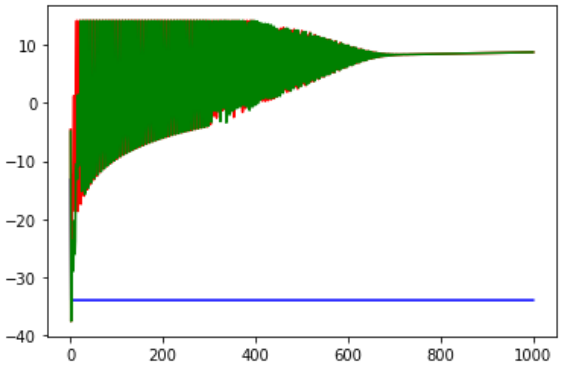
حل با ماتریس تصادفی غیر دوبل متصل و نا متغییر با زمان:

```
A = np.array([[1/2,1/4,1/4],[1/2,1/4,1/4],[1/2,1/4,1/4]])
```

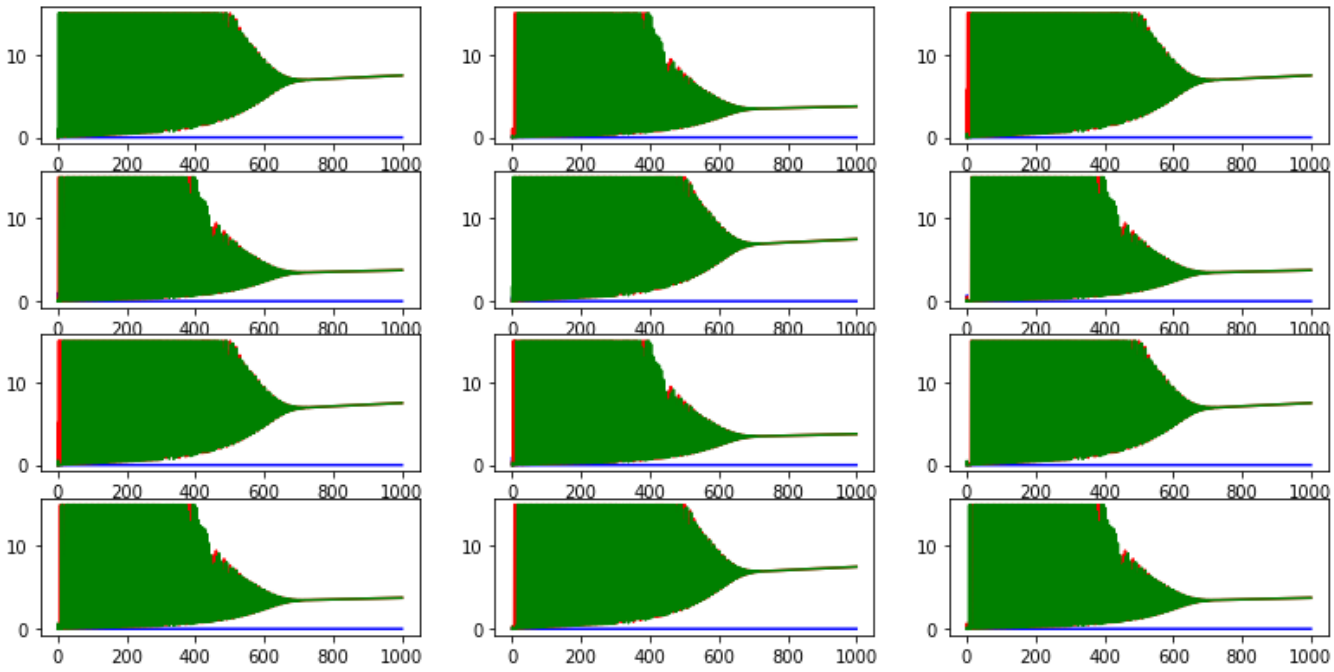
این شبیه سازی را برای 1000 ایتریشن انجام دادیم و حاصل موفقیت آمیز بود:

هر رنگ در هر نمودار نشان دهند مقدار هر نود برای آن نمودار است :

مقدار تابع ما به این صورت همگرا شدن برای هر سه نود در یک نمودار:



مقدار هر یک از 12 فلو برای هر نود به صورت زیر همگرا شده است:



همانطور که مشاهده می کنید الگوریتم به نطقه بهینه همگرا نشد.

برای بهینه سازی توزیع شده ماتریس تصادفی باید دوبل باشد.

حمید رضا کاشانی

پروژه دوم

810100441

حل با ماتریس تصادفی دوبل متصل متغییر با زمان:

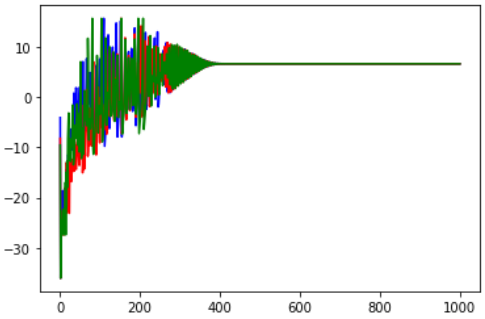
در هر ایتریشن به صورت دوره ای از این سه ماتریس استفاده شده است:

```
A1 = np.array([[1/2,1/4,1/4],[1/4,1/2,1/4],[1/4,1/4,1/2]])
A2 = np.array([[1/3,1/3,1/3],[1/3,1/3,1/3],[1/3,1/3,1/3]])
A3 = np.array([[0,1,0],[0,0,1],[1,0,0]])
```

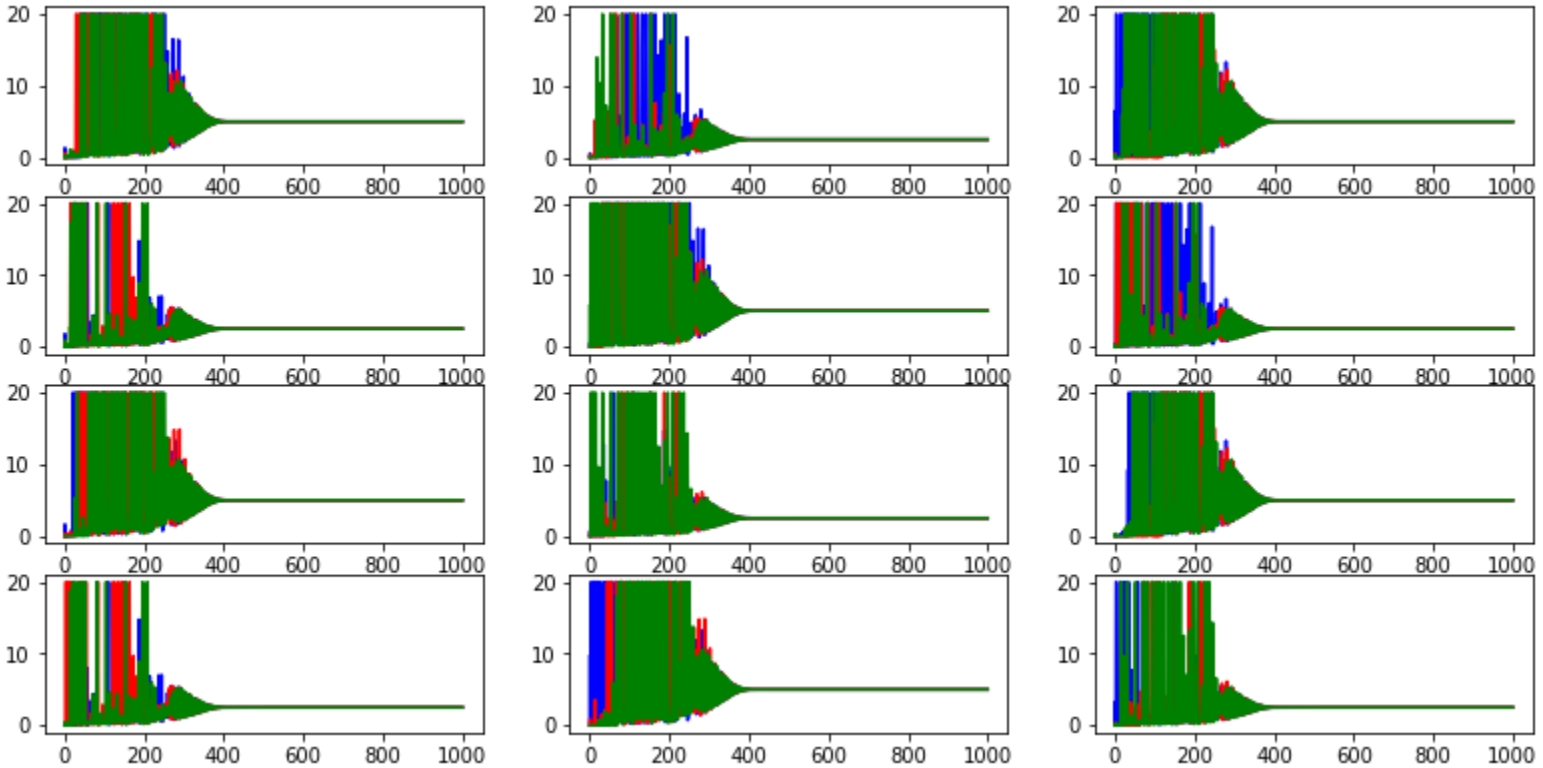
این شبیه سازی را برای 1000 ایتریشن انجام دادیم و حاصل موفقیت آمیز بود:

هر رنگ در هر نمودار نشان دهند مقدار هر نود برای آن نمودار است :

مقدار تابع ما به این صورت همگرا شدن برای هر سه نود در یک نمودار:



مقدار هر یک از 12 فلو برای هر نود به صورت زیر همگرا شده است:



حمید رضا کاشانی 810100441

پروژه دوم

حل با ماتریس تصادفی دوبل متصل دوره ای و متغییر با زمان:

در هر ایتریشن به صورت دوره ای از این سه ماتریس استفاده شده است:

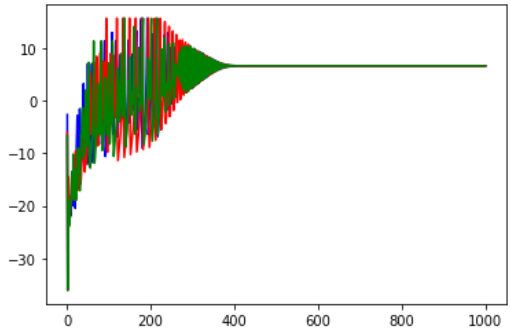
```
A1 = np.array([[1/2,1/2,0],[1/2,1/2,0],[0,0,1]])
A2 = np.array([[1/2,0,1/2],[0,1,0],[1/2,0,1/2]])
A3 = np.array([[1,0,0],[0,1/2,1/2],[0,1/2,1/2]])
```

در ابتدا نود سوم متصل نیست و سپس نود دوم و بعد نود اول به صورت پریودیک.

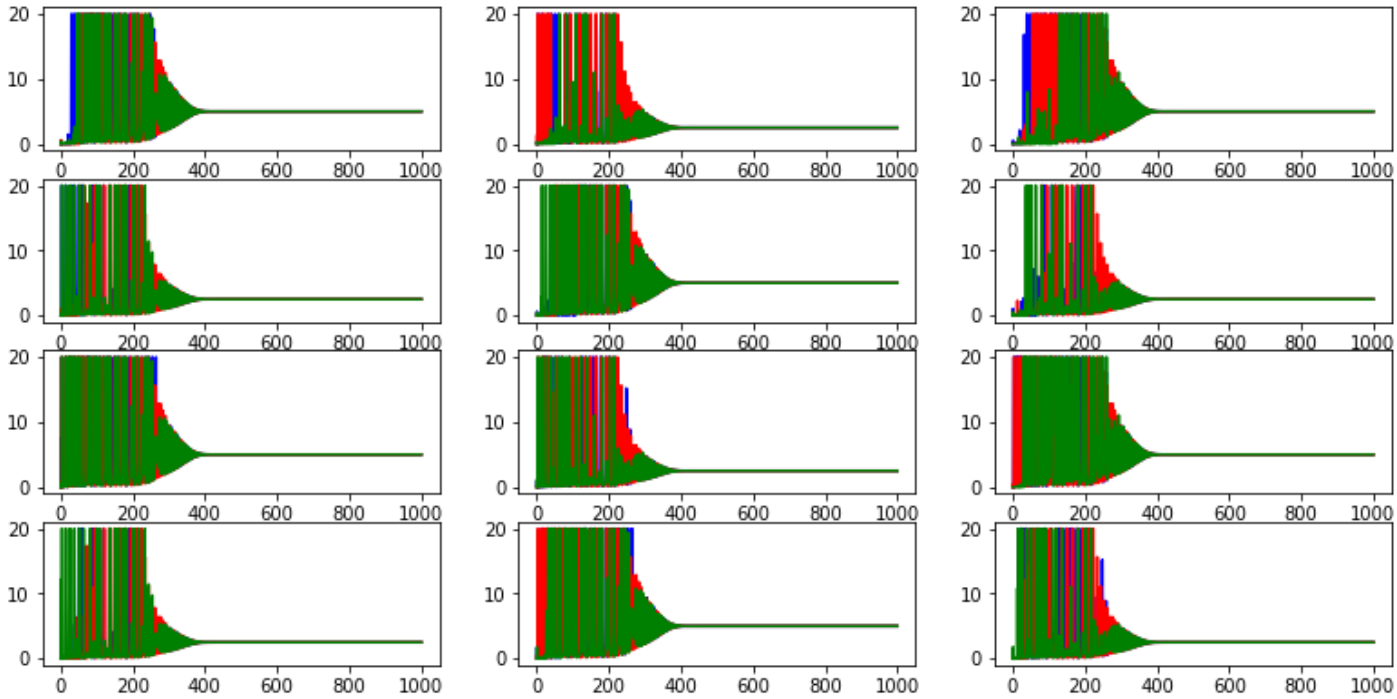
این شبیه سازی را برای 1000 ایتریشن انجام دادیم و حاصل موفقیت آمیز بود:

هر رنگ در هر نمودار نشان دهند مقدار هر نود برای آن نمودار است :

مقدار تابع ما به این صورت همگرا شدن برای هر سه نود در یک نمودار:



مقدار هر یک از 12 فلو برای هر نود به صورت زیر همگرا شده است:



حمید رضا کاشانی 810100441

پروژه دوم

نمودار متصل دوره ای مقدار اعوجاج و خارج شدن از مسیر همگرایش در طول مسیر بیشتر بود که کاملا طبیعی است و مشاهده می شود که زمان همگرایی هر سه ماتریس تصادفی به هم نزدیک است که این به نظر به خاطر کم بودن تعداد نود های ما است و اطلاعات به سرعت به همگی میرسد.