سیستم‌های توزیع‌شده

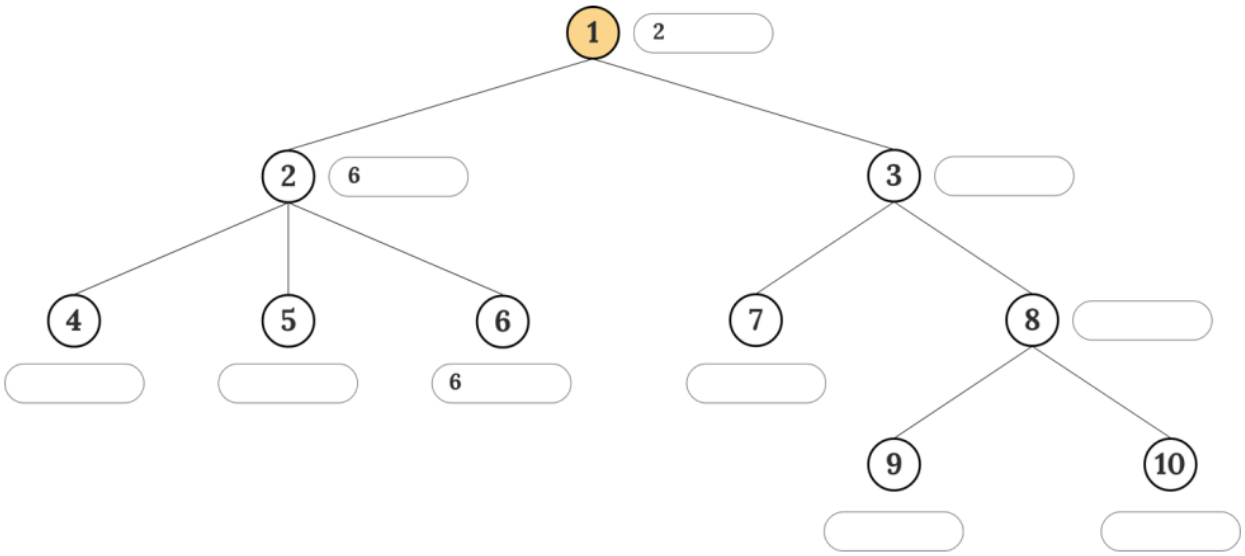
تمرین سری پنجم

دکتر کمندی

پارسا محمدپور

پرسش یک

شکل زیر یک حلقه ریموند را نشان می‌دهد. گره یک توکن را در اختیار دارد. اعداد کنار هر گره نشان‌گر صف کنونی هستند.



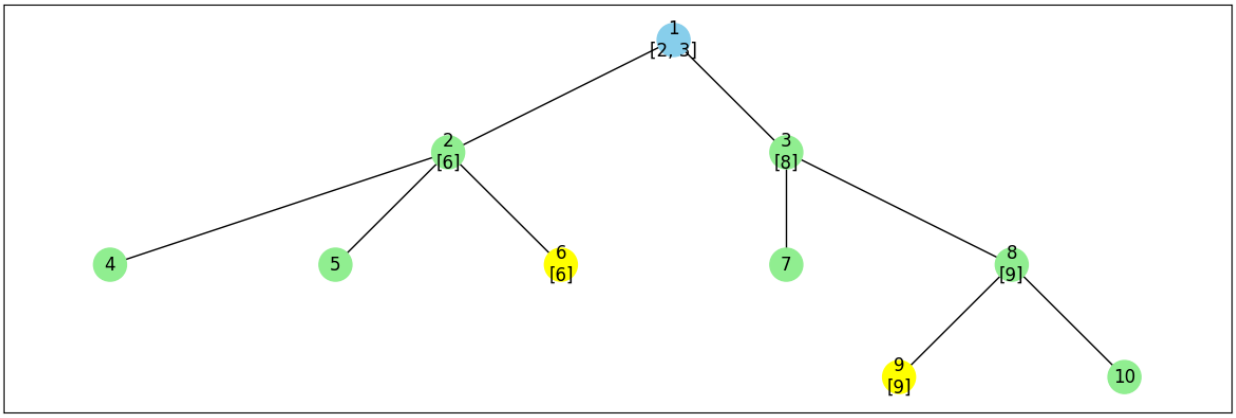
الف) فرض کنید در حالیکه گره یک توکن را همچنان در اختیار دارد، گره‌های نه، هفت و پنج به ترتیب درخواست ورود به ناحیه بحرانی بدهند. ورودی‌های صف‌های تمام گره‌ها را پس از پردازش درخواست این گره‌ها بدست آورید.

ابتدا یک توضیح کلی در رابطه با عملکرد الگوریتم ارائه می‌دهیم. این الگوریتم، یکی از انواع الگوریتم‌های مبتنی بر توکن[1] می‌باشد. در این الگوریتم، ما ابتدا باید یک **درخت** (دلیل استفاده از درخت اسن است که نباید در این الگوریتم دور وجود داشته باشد؛ چون در صورت وجود داشتن دور، امکان رخداد بن‌بست[2] بوجود می‌آید) از اتصال سایت‌ها داشته باشیم. تنها سایتی(گره‌ای) که توکن را در اختیار داشته باشد توانایی ورود به ناحیه بحرانی را دارد. هر سایت، پس از تمام شدن کارش با ناحیه بحرانی و پس از خروج از آن، توکن را به سایتی که در ابتدای صفش قرار دارد پاس می‌دهد و ساختار درختی که در این الگوریتم وجود دارد، عوض می‌شود؛ به طوریکه سایتی که قبل از آن توکن را در اختیار داشت، الآن، فرزند سایتی می‌شود که در حال حاضر گره را در اختیار دارد. حال با توجه به اسن توضیحات و مفروضات داده شده، به سراغ حل این مسئله می‌رویم. (برای این مسئله یک کد هم زده شده است که می‌توان آن را در این لینک، مشاهده کرد) حالا در این الگوریتم برای یک حالت خاص، می‌توانیم دو رویکرد مختلف را در پیش بگیریم.

در این وضعیت، سایت نه ابتدا درخواست خود را می‌دهد. پس باید در صف مربوط به خود سایت نه، مقدار نه وارد شود. سپس سایت هشت نیز درخواست را از سایت نه دریافت کرده و مقدار نه را در صف خودش جای می‌دهد و سپس درخواست را به سایت سه می‌دهد. سایت سه نیز پس از دریافت این درخواست، سایت هشت را در صف خود قرار می‌دهد و این درخواست را برای سایت یک

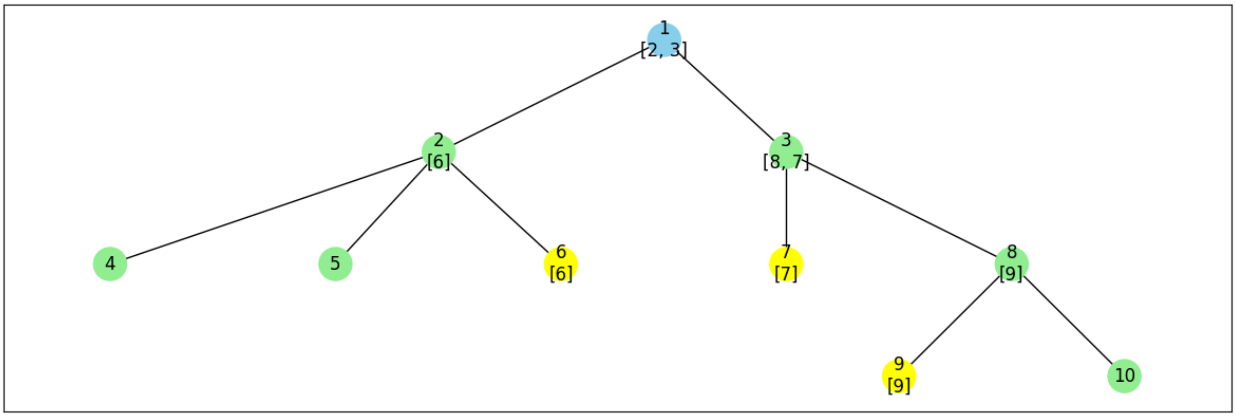
[1] Token-based algorithm
[2] Deadlock

ارسال می‌کند. سایت یک نیز پس از دریافت این درخواست، سایت سه را در صف خود اضافه می‌کند. سپس شکل سایت‌ها به همراه صف هایشان به صورت زیر می‌شود:



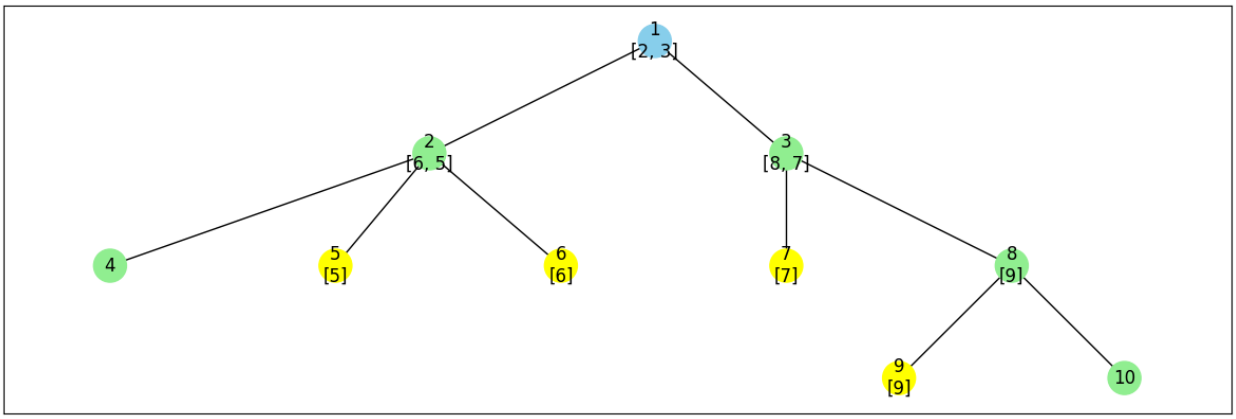
۱این شکل، گراف سایت‌ها و صف‌هایشان پس از دریافت درخواست سایت نه به وسیله همه سایت‌ها است. در این شکل سایتی که توکن را در اختیار دارد، با رنگ آبی، سایتی که درخواست توکن را دارد، با رنگ زرد و بقیه سایت‌ها هم با رنگ سبز نشان داده شده‌اند.

سپس طبق صورت سوال، سایت هفت درخواست خود را برای سایت سه ارسال می‌کند و سایت سه سایت هفت را در صف خود قرار می‌دهد سپس سایت سه نیز درخواست را برای سایت یک ارسال می‌کند و سایت یک نیز آن را دوباره در صف خود قرار می‌دهد. پس شکل گراف حاصل و صف‌های سایت‌ها به صورت زیر می‌شود:



۲شکل سایت‌هاو صف‌هایشان پس از دریافت درخواست سایت هفت

سپس سایت پنج درخواست خود را به سایت دو ارسال می‌کند و سایت دو علاوه بر اضافه کردنم سایت پنج، درخواست سایت پنج را به سایت یک نیز ارسال می‌کند. سپس سایت یک هم سایت دو را در صف خود اضافه می‌کند. پس شکل سایت‌ها و صف‌هایشان به صورت زیر می‌شود:

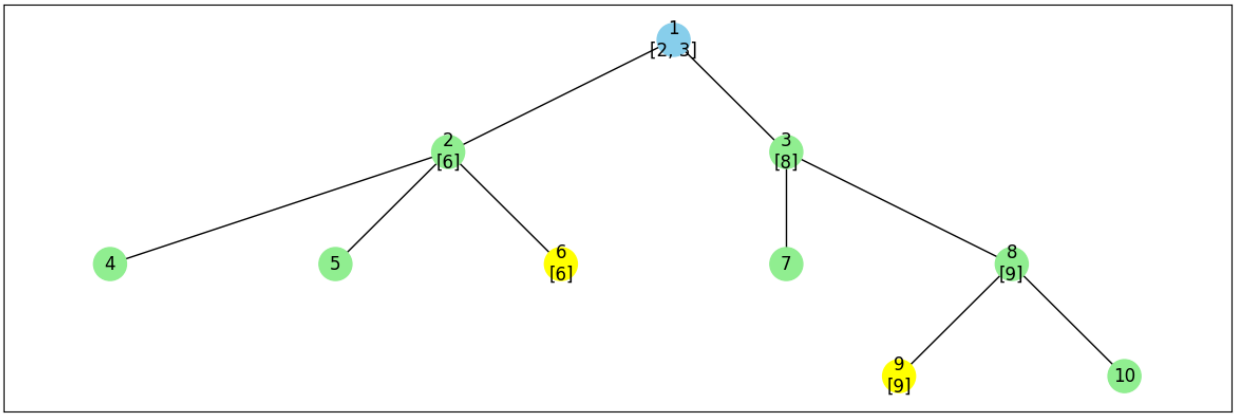


۳شکل سایت‌ها و صف‌هایشان پس از دریافت درخواست سایت پنج

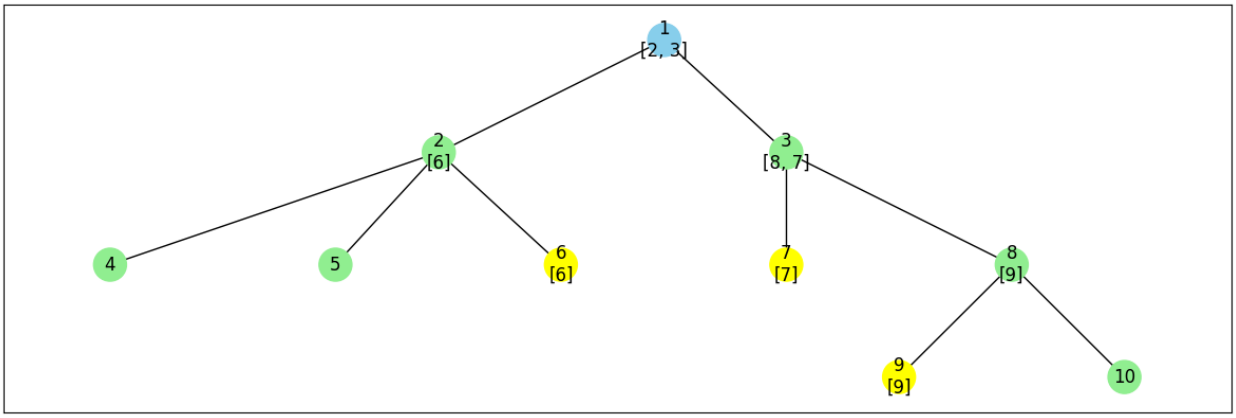
حالا در این قسمت با این چالش روبرو نشدیم، ولی حالا الآن وقتی که برای قسمت بعدی سایت یک می‌خواهد توکن را به سایت دو بدهد، چون در صف خودش چند سایت وجود دارد باید چه کند؟ دو رویکرد را می‌توانیم مدنظر قرار بدهیم. در یکی از این رویکردها، همیشه درخواست زودتر، زودتر به ناحیه بحرانی می‌رسد، در حالت دیگر درخواستی که نزدیک‌تر (نه دقیقا نزدیک‌تر،یعنی در همان نحیه باشد و نیازی نداشته باشد تا برای اینکه توکن به آن برسد از سایت بالایی رد شود) است، سریع‌تر به ناحیه بحرانی وارد می‌شود. برای مثال وقتی در این حالت، سایت یک می‌خواهد توکن را به سایت دو بدهد، این دو حالت را داریم: می‌توانیم برای سادگی سایت یک را در انتهای صف مربوط به سایت دو قرار دهیم و ادامه بدهیم(یعنی اگر سایتی که در حال واگذاری توکن است، صفش خالی نیست، خودش را در انتهای صف دیگری قرار دهد)؛ یا رااه دوم این است که با توجه به زمان منطقی درخواست سایت‌ها، آن را در جایی که سریع‌ترین درخواستش آمده است قرا دهیم, یعنی برای مثال اگر اولین درخواست موجود در سایت یک هنگام تحویل توکن به سایت دو، برای سایت سه و باید زودتر از درخواست سایت دو (آن درخواستی که برای سایت پنج است، نه آن درخواستی که برای سایت شش است) انجام شود، در این حالت اجام شود. در این حالت، ما می‌توانیم تضمین کنیم که همیشه درخواستی که زودتر آمده است، زودتر هم پردازش شود و توکن را در دست بگیرد.

هر دوی این حالات در کد آورده شده پیاده‌سازی شده‌اند. در این قسمت، نتایج حاصل از این دو تا رویکرد، تفاوتی نمی‌کرد و عکس‌هایی که تا اینجا قرار داده شده‌اند، برای حالتی است که صرفا به ته صف اضافه می‌کردیم.(یعنی حالت اول) ولی در این قسمت و قسمت‌های بعد، عکس‌های حاصل از پیاده‌سازی رویکرد دوم، که مبتنی بر زمان ارسال درخواست است، نیز آورده شده‌اند؛ اما دیگر توضیحی برایشان ارائه نمی‌شود.

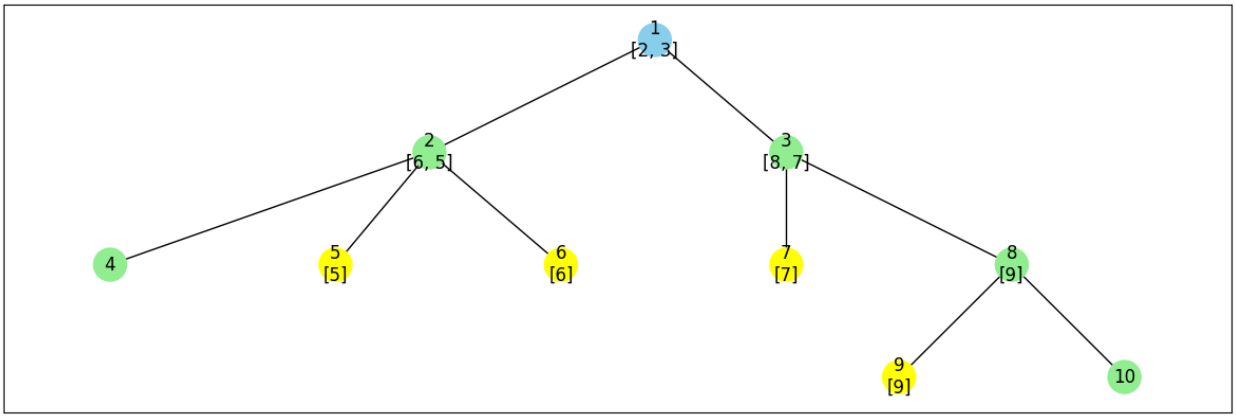
عکس‌های حاصل از پیاده‌سازی روشمبتنی بر زمان منطقی:



۴وضعیت سایت‌ها پس از پردازش درخواست سایت نه



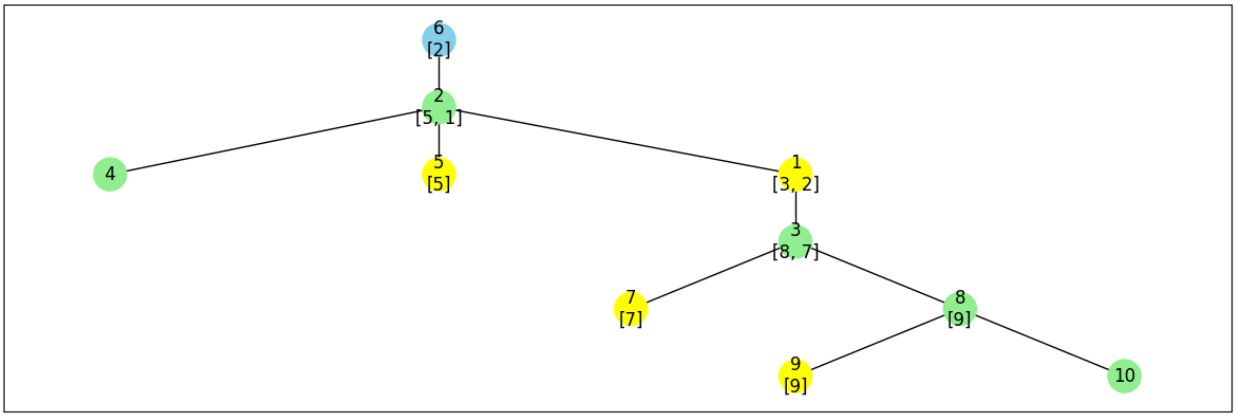
5وضعیت نهایی سایت‌ها پس از پردازش درخواست سایت ده



۶وضعیت نهایی سایت‌ها پس از پردازش درخواست سایت پنج

ب) فرض کنید درخواست دیگری برای ورود به ناحیه بحرانی از طرف گره‌ای ارسال نشود. ورودی‌های صف‌های گره‌های مرتبط را وقتی توکن به گره شماره شش می‌رسد، به دست آورید.

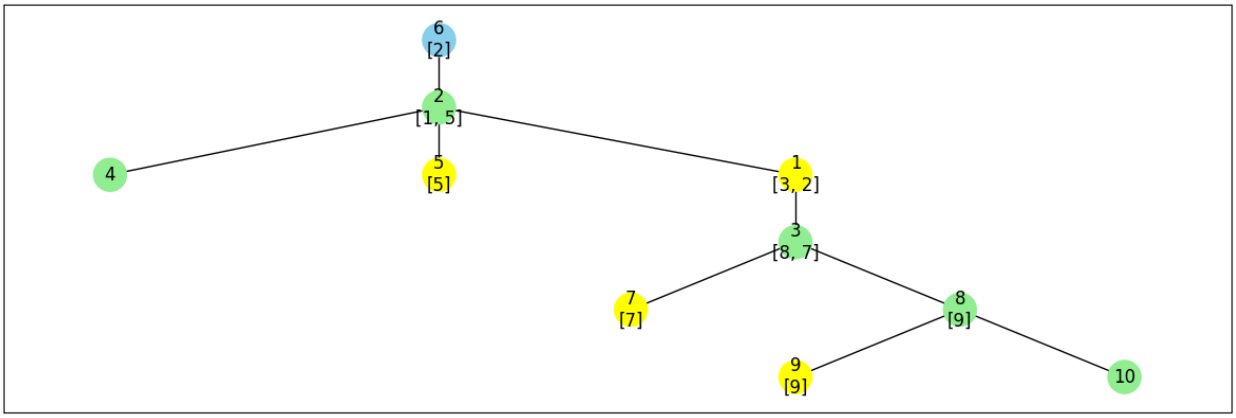
حالا با توجه به اینکه کار سایت یک با ناحیه بحرانی تمام شده‌است، پس از لیست خود اولین سایت را برمیدارد و توکن را به آن می‌فرستد و اما با توجه به اینکه درحال حاضر، صف آن، شامل تعدادی سایت هست همچنان، سایت دو مقدار سایت یک را نیز در انتهای صف خودش قرار می‌دهد. (حالت اول) سپس سایت دو نیز پس از دریافت توکن، با توجه به اینکه در ابتدای صفش، سایت شش قرار دارد، توکن را برای سایت شش ارسال می‌کند و با توجه به اینکه صف خودش خالی نیست، خودش (سایت دو) هم در انتهای صف مربوط به سایت شش، قرار می‌گیرد. پس در انتها، توکن در سایت شش قرار می‌گیرد و در صف سایت دو، سایت یک قرار دارد و در صف سایت شش هم، سایت دو قرار دارد. شکل حاصل به صورت زیر می‌شود:



۷شکل سایت‌ها و درخواست کننده‌ها که منتظر دیافت توکن هستند و سایت شش که رادای توکن است

پس بدین ترتیب، سایت شش توکن را دریافت می‌کند و در صف آن، سایت دو قرار دارد و در صف سایت دو، سایت‌های پنج و یک قرار دارند و به همین ترتیب سایت‌های موجود در صف سایت‌های دیگر هم مشخص شده‌اند.

حالا مطابق با توضیحات رائه شده در قسمت الف، نتیجه حاصل از پردازش درخواست‌ها مبتنی بر زمان ارسال درخواست‌شان را هم در ادامه قرار می‌دهیم.

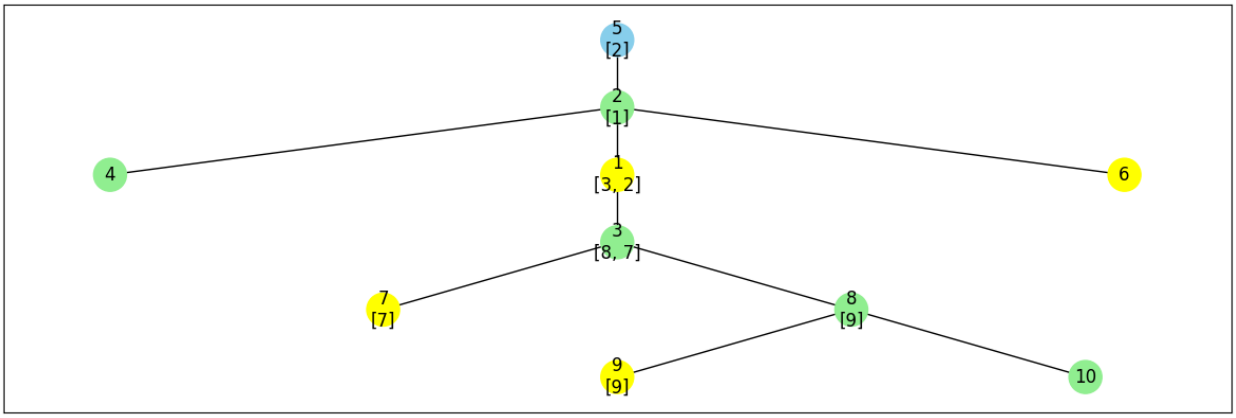


۸وضعیت نهایی سای‌ها پس از اتمام کار سایت یک و ارسال توکن برای سایت شش

ج) قسمت قبل را برای وقتی که توکن به گره نه می‌رسد، تکرار کنید.

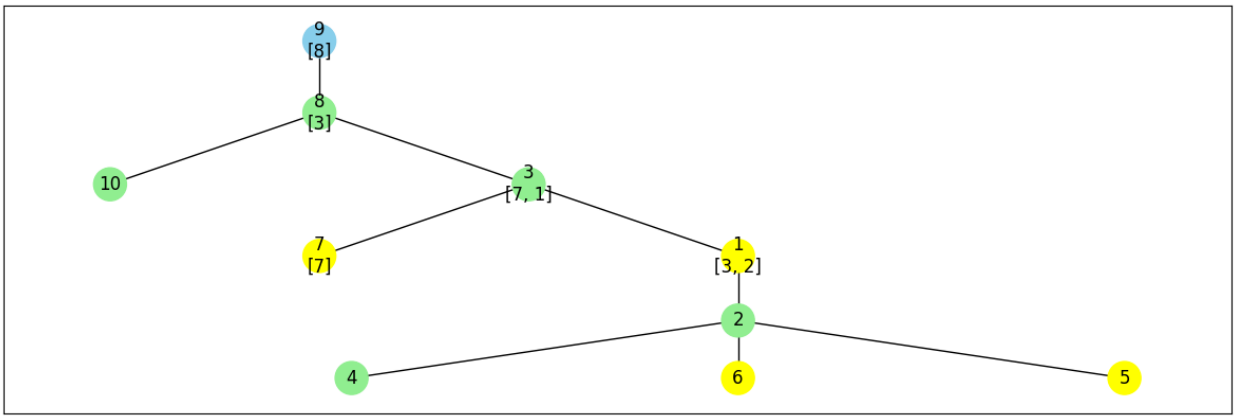
همین کار را به قدری انجام می‌دهیم تا توکن به گره شماره نه برسد. پس به ترتیب گفته‌ده جلو می‌رویم و مطابق الگوریتم مراحل را طی می‌کنیم.

در اولین مرحله، پس از اتمام کار سایت شش با ناحیه بحرانی، سایت شش توکن را انتقال می‌دهد تا به سایت پنج برسد. (چون ما صرفا بدون درنظر گرفتن زمان منطقی ارسال هر درخواست، سایتی که توکن را می‌دهد به ته صف سایت گیرنده اضافه می‌کنیم، سایت پنج، سایت بعدیست. درحالیکه اگر با زمان منطقی جلو میرفتیم، سایت بعدی باید سایت نه می‌بود) سایت پنج توکن را در اختیار دارد. شکل حاصل پس از انجام این مرحله و قرارگیری توکن در سایت پنج، به صورت زیر می‌شود:



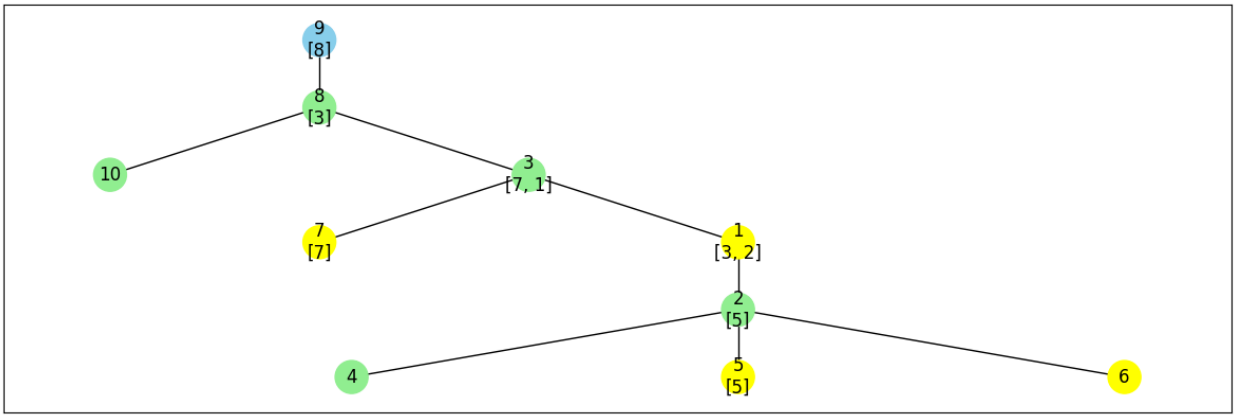
۹شکل سایت‌ها و صف‌هایشان. در این لحظه سایت پنج توکن را در اختیار دارد و می‌تواند وارد ناحیه بحرانی شود

سپس یک مرحله دیگر جلو می‌رویم. این سری توکن به سایت نه می‌رسد. چون اولویت با اولین سایتی است که در صف هر سایت قرار دارد هنگامی که آن سایت توکن را در اختیار بگیرد. شکل حاصل به صورت زیر می‌شود:



۱۰ شکل نهایی سایت‌ها و صف‌هایشان پس از در اختیار گرفتن توکن به وسیله سایت نه

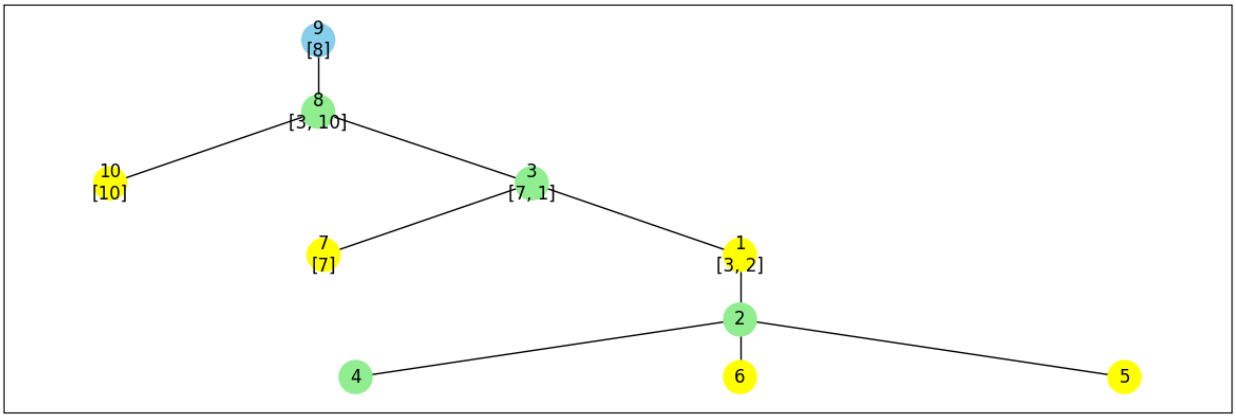
در این قسمت نیز مشابه با قسمت‌های پیشین، نتیجه حاصل از اجرای الگوریتم مبتنی بر زمان را هم قرار می‌دهیم. اما در این قسمت برخلاف قسمت‌های قبلی، نتیجه تغیر می‌کند و درخواست سایت نه، زودتر اجرا می‌شود.



۱۱ نتیجه الگوریتم مبتنی بر زمان پس از اتمام پردازش سایت شش و ارسال آن به سایت نه

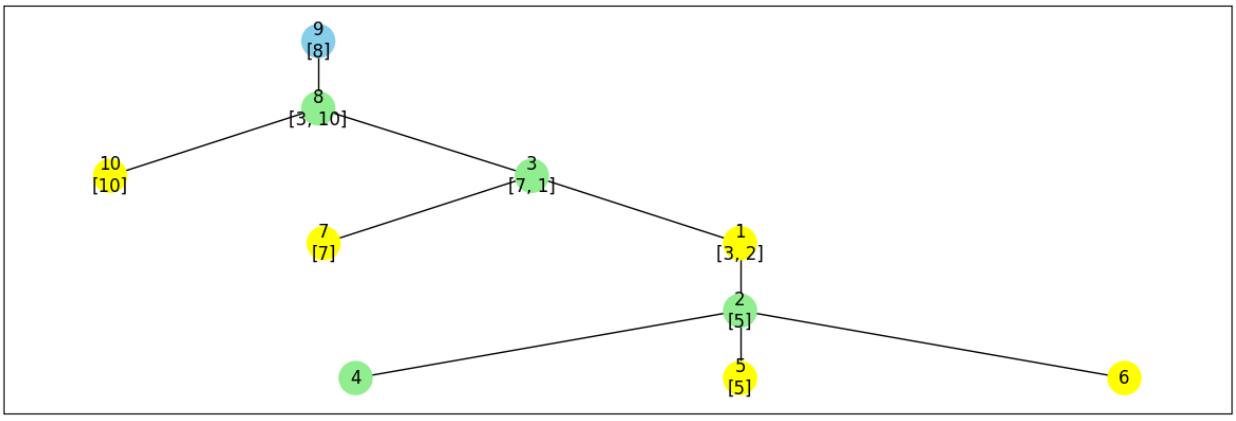
د) فرض کنید وقتی توکن در اختیار گره نه است، گره ده درخواست ورود به ناحیه بحرانی می‌دهد. ورودی‌های صف گره‌های مرتبط را بدست آورید.

در این حالت، سایت نه دارای توکن است و در صف خود سایت هشت را دارد. (سایت موجود در صف هر سایت و اطلاعات دیگر در شکل ۱۰ موجود است). سپس سایت ده می‌خواهد وارد ناحیه بحرانی شود. برای این موردذ درخواست خود را به سایت هشت ارسال می‌کند. (سایت هشت در حال حاضر در درخت این الگوریتم، پدر این سایت می‌باشد) سپس سایت هشت هم درخواست را برای سایت نه می‌فرستند. شکل نهایی به صورت زیر می‌شود:



۱۲شکل نهایی سایت‌ها پس از اینکه سایت ده درخواست ورود خود را ارسال کرد

در این قسمت نیز مشابه با قسمت‌های پیشین، نتیجه حاصل از اجرای الگوریتم مبتنی بر زمان را هم قرار می‌دهیم. اما در این قسمت برخلاف قسمت‌های قبلی، نتیجه تغیر می‌کند و درخواست سایت نه، زودتر اجرا می‌شود.



۱۳نتیجه ارسال درخواست توسط سایت ده برای سایت نه که دارنده توکن است

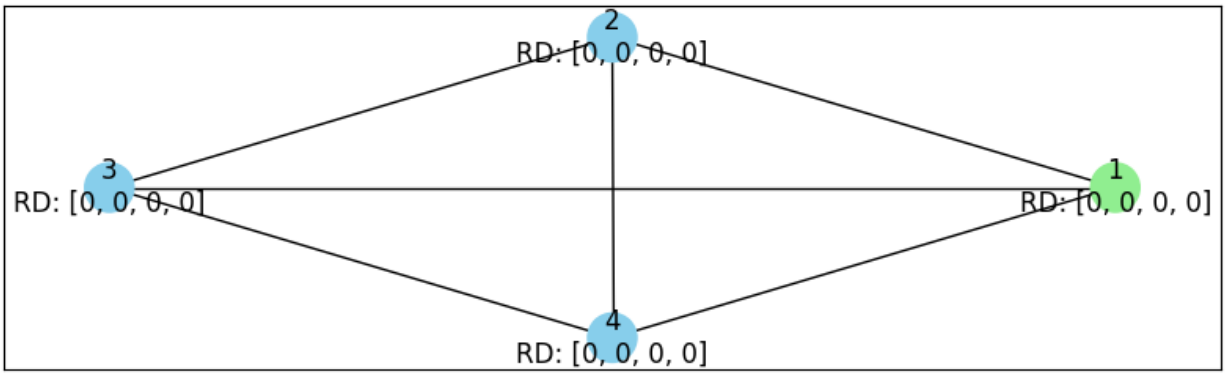
پرسش دو

گروهی از سیستم‌های توزیع‌شده P_1، P_2، P_3 و P_4 را در نظر بگیرید که یک شئی را به شاتراک می‌گذارند و برای مدیریت انحصار متقابل[1]، از الگوریتم ریکارت-آگراوالا[2] استفاده می‌کنند. P_1 در حال حاضر در ناحیه بحرانی است و هیچ گره دیگری در حالت "درخواست"[3] نیست. اکنون درخواست‌های P_4، P_2 و P_3 (به ترتیب) برای ورود به همان ناحیه بحرانی را در نظر بگیرید.

الف) وضعیت "درخواست"، "نگه‌داشته‌شده"[4] و ... موجود در صف هر سیستم را مشخص کنید.

ابتدا یک توضیح مختصری در رابطه با الگوریتم ریکارت-آگراوالا ارائه می‌دهیم. این الگوریتم به این صورت کار می‌کند که هر سایتی که درخواست ناحیه بحرانی را داشته باشد، درخواست خود برای ورود به این ناحیه بحرانی با مقدار زمان منطقی خودش را برای تمام سایت‌های دیگر ارسال می‌کند. سپس باید منتظر دریافت پیام جواب از طرف تمام سایت‌های دیگر باشد. همچنین در هنگام دریافت درخواست از سمت سایتی که درخواست ورود به ناحیه بحرانی را دارد، اگر اولویت خودش بیشتر باشد، یعنی یا زمان منطقی خودش در هنگام درخواست کمتر باشد یا شماره شناسه یکتایش[5] کمتر باشد، پیام جواب را برای آن سایت ارسال نمی‌کند و مقدار موجود یک متغیر برداری به نام آردی[6] در درایه[7] مطابق با شناسه یکتا آن سایت را برابر با یک قرار می‌دهد تا بتواند آن را حفظ کند و بعدا از آن استفاده کند تا بتواند بعد از به اتمام رسیدن کارش در ناحیه بحرانی، پیام جواب آن درخواست‌ها را ارسال کند. کدهای پایتون این قسمت (برخلاف قسمت قبلی بدون پیاده‌سازی الگوریتم) و صرفا برای شکل‌ها که به طور دستی می‌باشد، در این لینک قرار دارد.

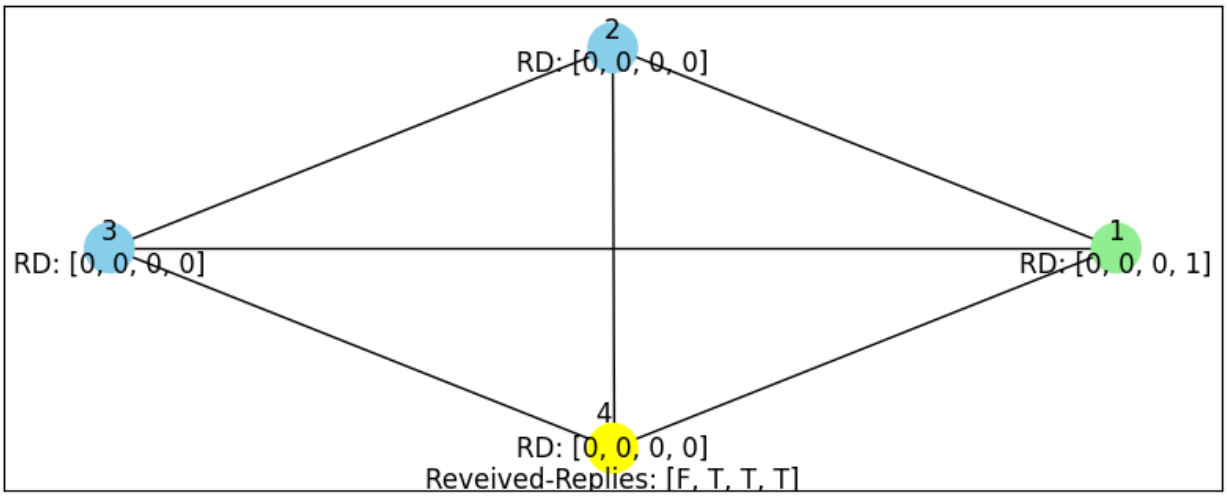
حال با توجه به اطلاعات داده شده درباره این الگوریتم و اطلاعات صورت سوال، به سراغ پیاده‌سازی و بدست آوردن اطلاعات پس از رسیدن پیام درخواست در این سایت‌ها می‌رویم. در این قسمت، طبق گفته صورت سوال، در ابتدا سایت یک در حال اجرای ناحیه بحرانی است و هیچ سایت دیگری درخواست را ارائه نداده است. پس شکل برداری آن‌ها به صورت زیر است.



شکل سایت‌های اولیه و بردارهای نگه‌دارنده متغیرهایشان (بردار آردی). همچنین رنگ سبز نشان‌دهنده این است که سایت در حال اجرای ناحیه بحرانی است. همچنین رنگ آبی نیز نمایانگر این است که آن سایت درخواستی مبنی بر ورود به ناحیه بحرانی نداده است. ۱۴

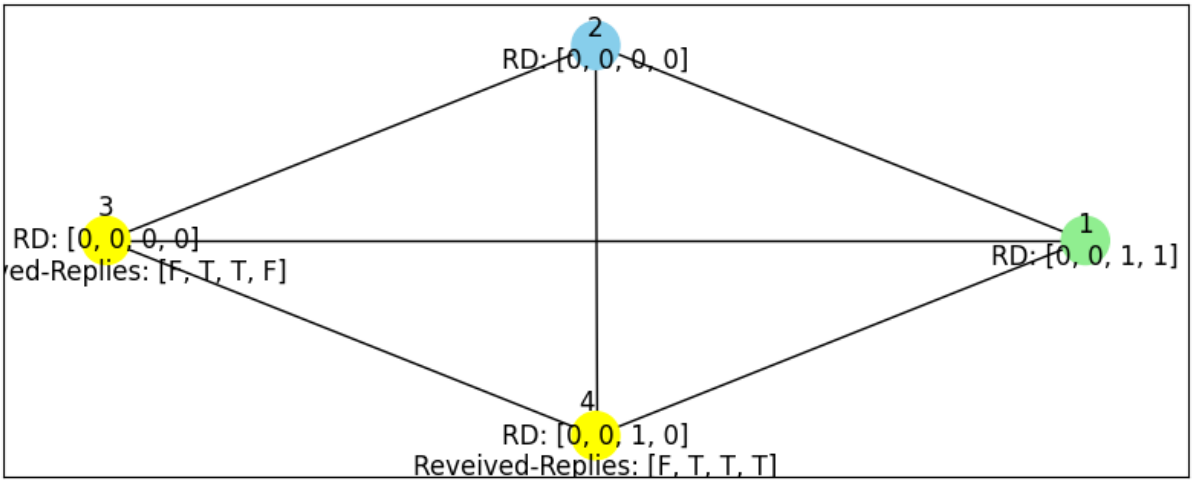
[1] Mutual Exclusion
[2] Ricarti-Agrawala
[3] Wanted
[4] Held
[5] UID
[6] RD
[7] index

سپس درخواستی از جانب سایت چهار ارسال می‌شود. پس همه سایت‌ها با توجه به اینکه سایت چهار اولین سایت ارسال کننده درخواست است، و خودشان درخواستی ندارند (درخواستشان هنوز نرسیده است یا در آینده خواهد رسید) پس به سایت چهار پیام جواب را می‌دهند به جز سایت یک؛ سایت یک چون در حال اجرای ناحیه بحرانی است، مقدار درایه متناظر با سایت چهار را برابر با یک قرار می‌دهد تا پس از اتمام کارش، به آن پیام جواب را ارسال کند. پس شکل سایت‌ها به صورت زیر می‌شود.



۱۵ شکل سایت‌ها و متغیر آردی آن‌ها پس از دریافت پیام سایت چهارم. رنگ زرد نشان‌دهنده اسن است که سایت مورد نظر در حال انتظار برای دریافت اجازه (جواب از تمام سایت‌های دیگر) برای ورود به ناحیه بحرانی است. همچنین بردار جواب‌های دریافتی[1] بیانگر این است که آیا پیام جواب از سمت دیگر سایت‌ها دریافت شده است یا خیر.

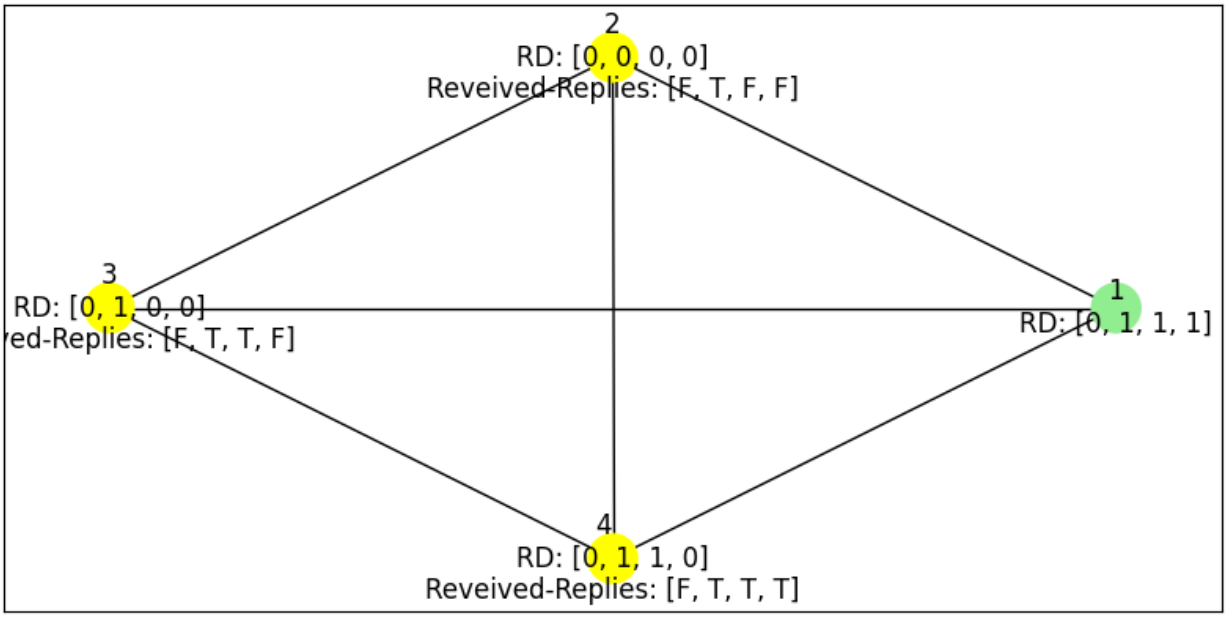
حالا به طور مشابه، پیام درخواست از سمت سایت سه ارسال می‌شود. دوباره طبق توضیحات ارائه شده جلو می‌رویم با این تفاوت که دیگر سایت چهار هم علاوه بر سایت یک، پیام جواب را برای سایت سه ارسال نمی‌کند. شکل زیر نشان‌دهنده وضعیت سایت‌ها و بردار آردی آن‌ها پس از دریافت پیام درخواست از جانب سایت سه می‌باشد.



۱۶شکل سایت‌ها و بردارهای آردی آن‌ها پس از دریفات پیام درخواست از جانب سایت سه

[1] Received Replies

حالا به طور مشابه، پیام درخواست از سمت سایت دو ارسال می‌شود. دوباره طبق توضیحات ارائه شده جلو می‌رویم با این تفاوت که دیگر سایت سه هم علاوه بر سایت یک و چهار، پیام جواب را برای سایت سه ارسال نمی‌کند. شکل زیر نشان‌دهنده وضعیت سایت‌ها و بردار آردی آن‌ها پس از دریافت پیام درخواست از جانب سایت سه می‌باشد.

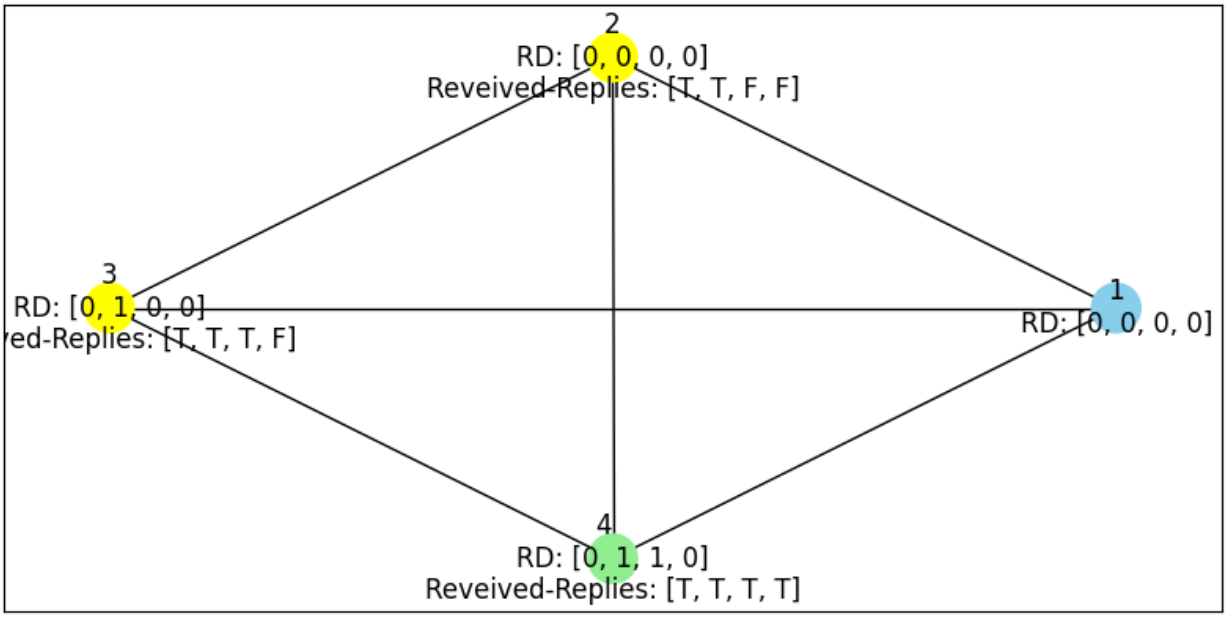


۱۷ شکل نهایی سایت‌ها پس از آنکه تمام سایت‌های گفته شده درخواست‌هایشان را به ترتیب گفته شده در صورت سوال ارسال کردند

پس شکل نهایی و بردار آردی نهایی هر سایت، به صورت آورده شده در شکل بالا می‌باشد. پس سایت یک، باید پس از اتمام کارش، به سایت‌های چهار، سه و دو پیام جواب را ارسال کند.

ب) اکنون P_1 از ناحیه بحرانی خارج شده و به تمام گره‌های مربوطه اطلاع می‌دهد که ناحیه بحرانی آزاد شده است. در این مرحله، وضعیت ورودی‌های صف را در هر سیستم نشان دهید.

طبق الگوریتم که در ابتدای قسمت قبل توضیح داده شده است، سایتی که در حال اجرای ناحیه بحرانی است، پس از اتمام کارش با ناحیه بحرانی، باید جواب تمام پیام‌های درخواستی‌ای که در خودش دارد، مقدار متغیر درایه متناظر با شناسه آن‌ها در بردار آردی برابر با یک است، ارسال کند؛ البته به شرط آن که درخواست بعدی خودش، در زمانی پس از درخواست آن‌ها آمده باشد. پس بدین ترتیب، سایت چهار، پس از دریافت پیام جواب از جانب سایت یک، چون تمام جواب‌های مورد نیاز را دریافت کرده است، پس به ناحیه بحرانی وارد می‌شود. پس شکل نهایی به صورت زیر می‌شود.



۱۸شکل سایت‌ها پس از اینکه سایت یک کارش با ناحیه بحرانی به اتمام رسید و پیام‌های جواب را برای سایت‌های دیگر ارسال کرد

پرسش سه

در خصوص احتمال رخداد بن‌بست در الگوریتم میکاوا[1] بحث کنید.

ابتدا به سراغ تعریف بن‌بست می‌رویم. تعریف بن‌بست به این صورت است که اگر حداقل یک فرآیند باشد که درخواست ورود به ناحیه بحرانی را داشته باشد (به اصطلاح در وضعیت تلاش برای ورود[2] باشد)، ولی هیچ فرآیندی موفق به ورود به منطقه بحرانی نشود. حال به سراغ توضیح الگوریتم میکاوو میرویم و سپس، به بررسی وقوع این پدیده در این الگوریتم میپردازیم .الگوریتم میکاوا به این صورت است که هر فرآیندی شامل یک مجموعه است که برای ورود به ناحیه بحرانی باید اجازه تمام فرآیندهای موجود در مجموعه‌ش را بگیرد، به اصطلاح از انواع الگوریتم‌های مبتنی بر حدنساب[3] است. هر مجموعه، در صورت دریافت یک پیام درخواست[4]، اگر قبلا اجازه را به کسی نداده باشد، اجازه ورود به آن ناحیه بحرانی را به آن می‌دهد، دز غیر این صورت آن درخواست را در یک صف نگه میدارد و به محض اینکه پیام آزادسازی از سمت فرآیندی که در آن لحظه، آن را گرفته بود دریافت کند، پیام جواب[5] را به فرآیندی که پیامش در ابتدای صفش بود، می‌دهد .در این الگوریتم امکان وقوع بن‌بست وجود دارد. برای اثبات این مورد، با توجه به مثال آورده شده در اسلاید درس، یک رخداد را نشان می‌دهیم که در آن بن‌بست بوجود بیاید. در این حالت فرض میکنیم که گره (سایت) یک درخواست بکند و همچنین در همین حال، سایت‌های دو و پنج هم درخواست بکنند. با توجه به تصویر زیر، که از اسلاید درس آورده شده است، میدانیم که سایت یک باید از سایت‌های یک، دو، سه و چهار درخواست بکند و اجازه آن‌ها را بگیرد و همچنین سایت‌ها دو هم باید از سایت‌های دو، پنج و هشت درخوسات بکند، سایت پنج هم باید از سایت پنج، یک، شش و هفت درخواست بکند و اجازه بگیرد. با این حساب، سایت یک به خودش اجازه میدهد چون درخواستش زودتر به خودش میرسد، سایت دو هم به خودش اجازه می‌دهد چون درخواستش زودتر به خودش رسیده است و سایت پنج هم به خودش اجازه میدهد چون درخواستش زودتر به خودش رسیده است. حالا ما در اینجا و در این حالت به بن‌بست میخوریم. چرا؟ چون الآن سایت یک، نیاز دارد تا از سایت دو اجازه بگیرد، ولی نمیتواند چون سایت دو به خودش اجازه ورود داده است، همچنین سایت دو هم نمیتواند وارد منطقه بحرانی شود، چون باید اجازه سایت پنج را بگیرد که ممکن نیست چون سایت پنج به خودش اجازه داده است و به کسی فعلا اجازه نمیدهد، سایت پنج هم نمیتواند وارد منطقه بحرانی شود به دلیل اینکه نیازمند دریافت اجازه سایت یک دارد، ولی نمیتواند آن را بگیرد چون سایت یک به خودش قبلا اجازه داده است. پس در این حالت به مشکل بن‌بست میخوریم.

---

[1] Maekawa
[2] Trying
[3] Quorum
[4] Request
[5] Reply