سوال ۱: اثبات

بردار ساعت[1]، آخرین زمان محلی دیده‌شده در گره[2]ای (یا فرآیند[3]های) دیگر را نگه‌داری می‌کند. برای این‌که پیش‌رفت هر گره توسط گره‌ی دیگر قابل پی‌گیری و ردیابی باشد، یک بردار با اندازه n مورد نیاز است. هم‌چنین چون این رابطه‌ی strong consistency بین گره‌های برقرار است، پس داریم:

$C(e_i) < C(e_j) \Rightarrow e_i -> e_j$

بنابراین برای به‌روز نگه‌داشتن و برقراری رابطه‌ی علیت بین دو رخداد، باید راجع‌به n -1 گره‌ی دیگر (زمان محلی آن‌ها) اطلاعات داشته باشد.

[1] Vector clock
[2] Node
[3] Process

سوال ۲: NTP Server

در سیستم‌های متمرکز، نیازی به هماهنگی ساعت نیست، زیرا در کل تنها یک ساعت وجود دارد. فرآیندها تنها با صادر کردن یک فراخوانی سیستمی، زمان را دریافت می‌کنند. وقتی یک فرآیند دیگری بعد از آن تلاش می‌کند که زمان را دریافت کند، مقدار زمان بیش‌تری را دریافت می‌کند. بنابراین، در چنین سیستم‌هایی یک ترتیب واضحی از روی‌دادها وجود دارد و هیچ ابهامی در زمانی که روی‌دادها رخ داده‌اند، وجود ندارد (به دلیل وجود نداشتن تداخل و افزایشی بودن مقدار زمان). در سیستم‌های توزیع‌شده، یک ساعت سراسری یا حافظه‌ی مشترک وجود ندارد. هر پردازنده، ساعت درونی خود را دارد و برداشت خود را از زمان دارد؛ به بیان دیگر این برداشت بین همه‌ی پردازنده‌ها (گره‌ها) یک‌سان نیست. در عمل، این ساعت‌ها می‌توانند به‌راحتی چند ثانیه در روز از هم دور شوند (ناهماهنگی رخ دهد) که در طول زمان منجر به خطاهای قابل‌توجهی می‌شود؛ هم‌چنین، از آن‌جایی که ساعت‌های مختلف با نرخ‌های متفاوتی تیک می‌زنند، ممکن است همیشه هم‌گام‌سازی نشوند، اگرچه ممکن است هنگام شروع هم‌گام‌سازی شوند. این به وضوح مشکلات جدی را برای برنامه‌هایی ایجاد می‌کند که به یک برداشت یکسان از زمان نیاز و وابستگی دارند.

در بیش‌تر برنامه‌ها و الگوریتم‌هایی که در سیستم‌های توزیع‌شده اجرا می‌شوند، باید مفهوم زمان را در یکی از زمینه‌های زیر بدانیم:

- زمانی از روز که یک رخ‌دادی روی یک ماشین خاص در شبکه اتفاق افتاده است.
- بازه‌ی زمانی بین دو رخ‌داد که روی ماشین‌های مختلفی در شبکه اتفاق افتاده است.
- ترتیب نسبی رویدادهایی که در ماشین‌های مختلف در شبکه اتفاق افتاده است.

به درخواست[4]های زمان‌محور در این ماشین‌ها نمی‌توان جواب داد، مگر این‌که یک برداشت مشترکی از زمان وجود داشته باشد. هم‌گام‌سازی ساعت یک فرآیندی است که تضمین می‌کند که پردازنده‌های توزیع‌شده‌ی فیزیکی یک برداشت مشترکی از زمان دارند. این کار تاثیر به‌سزایی در بسیاری از کاربردها، مانند امن‌کردن سیستم‌ها، عیب‌یابی، ریکاوری، انجام کارهای برنامه‌ریزی شده و مدون، سیستم‌های پایگاه‌داده و ... دارد. ار رایج است که برنامه‌های توزیع‌شده و پروتکل‌های شبکه از timeoutها استفاده کنند و عمل‌کرد آن‌ها بستگی به این موضوع دارد که پردازنده‌های پراکنده (به‌طور فیزیکی) تا چه اندازه هماهنگ شده‌اند. طراحی چنین برنامه‌هایی با هم‌گام‌سازی ساعت‌ها ساده‌سازی می‌شود. با توجه به نرخ‌های مختلف ساعت، ساعت‌ها در

[4]Query

مکان‌های مختلف ممکن است با زمان واگرا شوند و به‌طور دوره‌ای باید یک هم‌گام‌سازی ساعت انجام شود تا این انحراف ساعت در سیستم‌های توزیع‌شده اصلاح شود.

NTP یک پروتکل اینترنتی است که ساعت‌های کامپیوترهای مختلف را در سیستم‌های توزیع‌شده هم‌گام‌سازی کند. اصطلاح NTP هم بر پروتکل و هم به برنامه‌ی متشری·خدمت‌گزار که روی کامپیوترها اجرا می‌شوند، دلالت دارد. نحوه‌ی کار NTP شامل سه مرحله است:

۱. کاربر NTP یک درخواست برای تبادل زمان را با کارگزار آغاز می‌کند.

۲. سپس کاربر می‌تواند تاخیر ارتباط و آفست محلی آن را محاسبه کند و ساعت محلی خود را مطابق با ساعت کارگزار تنظیم کند.

۳. به‌عنوان یک قاعده، برای تنظیم اولیه‌ی ساعت، شش تبادل در یک بازه‌ی زمانی حدودا ۵ تا ۱۰ دقیقه‌ای نیاز است.

یک‌بار که هم‌گام شوند، کاربر ساعت را یک‌بار در هر ۱۰ دقیقه به‌روزرسانی می‌کند (که معمولا تنها نیاز به رد و بدل یک پیام دارد).

دستگاه‌های شبکه می‌توانند از کارگزارها نظرسنجی کنند و به رخدادهای و انتشار[5] NTP گوش می‌دهند تا اطلاعات را به موقع دریافت کنند.

به‌طور کلی دو نوع روش برای به‌دست‌آوردن اطلاعات زمان در NTP وجود دارد:

۱. انجمن‌های NTP مبتنی‌بر نظرسنجی[6]: رایج‌ترین حالت‌های ارتباط مبتنی بر نظرسنجی، حالت مشتری و حالت فعال متقارن هستند. آن‌ها درجه بالایی از دقت و قابلیت اطمینان را برای زمان‌بندی ارائه می‌دهند.

با حالت مشتری، دستگاه‌های شبکه به میزبان‌هایی اختصاص داده می‌شوند که زمان سرویس‌دهی می‌کنند و آن‌ها برای زمان درست نظرسنجی می‌کنند. سپس یک میزبان را برای هم‌گام‌سازی انتخاب می‌کند و هیچ اطلاعاتی را به خدمت‌گزار ارائه نمی‌دهد. این روی‌کرد برای مشتری‌هایی مانند سرورهای فایل و ایستگاه‌های کاری که با سایر مشتری‌ها هم‌گام نیستند، بهترین است.

با حالت فعال متقارن، دستگاه برای زمان درست از میزبان خود نظرسنجی می‌کند. هم‌چنین به نظرسنجی‌های میزبان خود پاسخ می‌دهد که اطلاعات مربوط به زمان را از دستگاه‌های شبکه جمع‌آوری

[5] Broadcast
[6] Polling

می‌کند. این حالت زمانی بهترین کار را انجام می‌دهد که چندین سرور با استفاده از مسیرهای مختلف شبکه به هم متصل باشند.

۲. انجمن‌های NTP مبتنی‌بر انتشار: انجمن های NTP مبتنی بر انتشار تا حدودی دقت و اعتماد کم‌تری نسبت به موارد مبتنی بر نظرسنجی دارند. آنها برای شبکه‌های محلی با پهنای باند، حافظه یا منابع واحد پردازش مرکزی[7]، محدود مناسب هستند.

در حالت مبتنی بر پخش، یک دستگاه شبکه به بسته‌های انتشار NTP که سرورهای زمان پخش را ارسال می‌کنند، گوش می‌دهد. اطلاعات زمانی فقط در یک جهت جریان دارد.

[7] Central Processing Unit (CPU)

سوال ۳: اثبات

- **بخش اول**

هر گره‌ای زمان محلی خود را می‌داند و زمانی که پیامی را دریافت می‌کند، تنها زمان مربوط به خود را به‌روزرسانی می‌کند. در هر لحظه از زمان، مقدار درایه‌ی iام، گره‌ی i برابر است با بیشینه مقداری که از قبل می‌داند و مقداری که در زمان دریافت پیام‌ها از دیگر گره‌ها دریافت می‌کند. برای هر یک از گره‌ها در واحد زمان، چون این مقدار افزایشی است (به دلیل به‌روزرسانی مقدار آن با بیشینه)، بنابراین مقدار نهایی آن برابر است با بشینه‌ی مقدار در تمامی حالات. با توجه به حالات ذکر شده، از کنار هم قرار دادن این زمان‌ها، حتما به مقدار بیشینه‌ی تمامی بردارها می‌رسیم، زیرا هر درایه‌ی آن دیگر درایه‌ها را قبلا و یا در حال حاضر مد نظر قرار داده و لزوما بیشینه‌ی آن مورد است.

- **بخش دوم**

در این قسمت باید به اثبات طرف چپ به راست و طرف راست به چپ بپردازیم:

- **اثبات طرف چپ به راست:** برای اثبات بخش اول (از چپ به راست عبارت شرطی) استدلالی مشابه بخش اول خواهیم داشت. یعنی روی‌داد داخلی (روی همان گره) و یا رویداد خارجی (روی گره‌ی دیگر) رخ داده است و در نهایت مسیری وجود داشته که e_i از آن طریق به e_j رسیده و اندیس i را در بردار به‌روزرسانی کرده است. اگر این اتفاق نمی‌افتاد حتما این مقدار در بردار خودش بزرگ‌تر می‌بود و در نتیجه رابطه‌ی علی که به صورت فرض در نظر گرفته بودیم نقض می‌شد و به تناقض می‌رسیدیم.
- **اثبات طرف راست به چپ:** اگر جایی این نابرابری وجود داشت نتیجه بگیریم رابطه علی برقرار بوده است. بالاخره نقطه‌ای وجود داشته که i قبل از j رخ داده و اندیس مدنظر در وکتور آن را به‌روزرسانی کرده است. پس اگر روابط علی را به شکل یک گراف پشت این رخداد رسم کنیم، قطعا از نقطه‌ای به ادامه‌ی آن متصل شده است که یعنی رابطه‌ی علی برقرار است.

در نتیجه‌ی این دو بخش می‌توان نتیجه گرفت عبارت دو شرطی مذکور برقرار است.