بهبود مسیریابی در داده‌ساختارهای توزیع‌شده‌ی مبتنی بر لیست‌های پرشی[1]

جواد شاهپریان* سید ایمان میررضایی‡ محمد قدسی§

چکیده

در این مقاله ما به ارائه‌ی یک الگوریتم جدید و کارآمد برای مسیریابی در شبکه‌های P2P می‌پردازیم. این الگوریتم به‌منظور جست‌وجو در شبکه‌های رویه‌ایِ مبتنی بر لیست‌های پرشی طراحی شده است و امکان جست‌وجوی بازه‌ای روی داده‌های چندبُعدی را نیز فراهم می‌کند. این الگوریتم بهبودیافته‌ی الگوریتم مسیریابی در داده‌ساختار SkipTree می‌باشد که ما میزان بهبود مسیریابی در الگوریتم پیشنهادی را به صورت نظری اثبات می‌کنیم. در این الگوریتم تعداد گام‌های مسیریابی از $O(\log_2 N)$ به $O(\log_r D)$ کاهش یافته، که D برابر طول مسیر و r هم پارامتر بهبود مسیریابی می‌باشد. همچنین در این الگوریتم اندازه جدول مسیریابی و تعداد اشاره‌گرهای ذخیره شده در آن نیز کاهش یافته است. از ویژگی‌های دیگر این الگوریتم این است که پارامترهای دیگر شبکه از قبیل حافظه مورد نیاز و توازن بار را تحت تأثیر قرار نمی‌دهد و سربار خاصی را به شبکه تحمیل نمی‌کند. به‌طور کلی هزینه مسیریابی در این روش مسیریابی نسبت به نمونه‌های مشابه بهبود قابل ملاحظه‌ای یافته است.

کلمات کلیدی

شبکه‌های P2P، الگوریتم‌های مسیریابی، لیست پرشی، داده‌ساختارهای توزیع‌شده، جست‌وجوی بازه‌ای

Routing Improvement in Skip List Based Distributed Data Structures

Abstract

This paper presents an efficient routing algorithm for Peer-to-Peer networks which is designed for searching in Skip List based overlay networks and supports range queries over multidimensional data. The proposed algorithm is improved version of SkipTree routing algorithm and the degree of improvement is proved formally. Using this algorithm, the number of hops will decrease from $O(\log_2 N)$ to $O(\log_r D)$ that D is distance between source and destination and r is routing improvement factor. In addition, size of routing table and number of stored pointers has been reduced. Furthermore we proved that this algorithm does not affect any other network parameters such as memory requirement, load balancing, and processing cost.

Keywords

P2P networks, Routing algorithms, Skip List, Distributed Data Structures, range query

[1] این مقاله تحت قرارداد پژوهشی شماره ۵۰۰/۸۴۷۱/ت مورخ ۸۵/۷/۴ از پشتیبانی معنوی و مادی مرکز تحقیقات مخابرات ایران بهره‌مند شده‌است.

*دانشجوی کارشناسی ارشد، دانشگاه صنعتی شریف، دانشکده‌ی مهندسی کامپیوتر، shahparian@ce.sharif.edu

‡دانشجوی کارشناسی ارشد، دانشگاه صنعتی شریف، دانشکده‌ی مهندسی کامپیوتر، mirrezaei@ce.sharif.edu

§استاد، دانشگاه صنعتی شریف و مرکز تحقیقات فیزیک نظری و ریاضیات(IPM)، ghodsi@sharif.edu

۱ مقدمه

شبکه‌های P2P به‌دلیل ویژگی‌ها و کاربردهای منحصر به فردی که دارند، بسیار مورد توجه قرارگرفته‌اند. در همین راستا، داده‌ساختارها و توپولوژی‌های متنوعی برای این شبکه‌ها ارائه شده است. SkipTree [?] از جمله داده‌ساختارهای توزیع‌شده است که مبتنی بر لیست‌های پرشی می‌باشد و مشکلات استفاده از جدول‌های درهم ساز را ندارد و به‌دلیل این که امکان ذخیره داده‌ها با کلید چندبُعدی را دارد و نیازی به توزیع یکنواخت ندارد مورد توجه قرار گرفته است. ما با ایجاد تغییراتی در نحوه ایجاد اشاره‌گرها و اتصال‌های میان گره‌ها و همچنین ایجاد تغییراتی در نحوه ساخت جدول‌های مسیریابی در SkipTree مسیریابی آن را بهبود می‌دهیم. هزینه مسیریابی با الگوریتم پیشنهادی ما به $O(\log_r D)$ کاهش می‌یابد که در آن D طول مسیر و r فاکتور میزان بهبود مسیریابی در شبکه است. در ادامه در بخش ۲ به مروری بر ساختار SkipTree پایه می‌پردازیم، در بخش ۳ الگوریتم مسیریابی بهبودیافته را معرفی می‌کنیم و در بخش ۴ به تحلیل نظری الگوریتم پیشنهادی می‌پردازیم، در پایان هم در بخش ۵ به بیان نتیجه کارهای انجام شده می‌پردازیم.

۲ مروری بر ساختار SkipTree پایه

SkipTree یک داده‌ساختار توزیع‌شده‌ی متوازن در شبکه‌های P2P است که به‌منظور ذخیره داده با کلید چندبُعدی طراحی شده است. منظور از توازن در این ساختار، توزیع یکنواخت بار در کل شبکه است. از ویژگی‌های دیگر این ساختار امکان جست‌وجوی بازه‌ای می‌باشد که می‌تواند همانند جست‌وجوی نقطه‌ای آن را در تعداد $O(\log N)$ گام انجام دهد. ساختار SkipTree یک ساختار کاملا غیرمتمرکز است که در آن هر گره به $O(\log N)$ گره دیگر متصل است. حافظه مورد استفاده برای ایجاد اتصال‌ها در هر گره به‌طور متوسط برابر $O(\log N \log\log N)$ و در بدترین حالت برابر $O(\log^2 N)$ است.

این داده ساختار توزیع‌شده از دو بخش اصلی تشکیل می‌شود، یکی درخت افراز که برای تقسیم فضای جستجو میان گره‌ها به کار می‌رود و دیگری اتصال‌های میان گره‌ها که از روشی شبیه ساختار لیست پرشی استفاده می‌شود. در این ساختار درخت افراز توزیع شده اقدام به تقسیم فضا می‌کند و فضا را به ناحیه‌های کوچک‌تری تقسیم می‌کند که هر گره در شبکه مسئول یکی از این ناحیه‌ها می‌باشد. در SkipTree برگ‌های درخت، گره‌های شبکه هستند و گره‌های میانی درخت معادلات صفحه هستند که برای تقسیم فضا به کار می‌روند. نقش اصلی درخت افراز ایجاد یک ترتیب کلی بین ناحیه‌ها می‌باشد. نحوه مسیریابی و ایجاد اتصال‌ها بین گره‌هامستقل از شکل درخت افراز است و به همین دلیل نیازی نیست که درخت افراز یک درخت متوازن باشد. برگ‌های درخت هم به‌صورت لیست پرشی به هم متصل هستند و یک ترتیب کلی از چپ به راست بین آن‌ها حاکم است. ویژگی دیگر SkipTree پشتیبانی از جست‌وجوی بازه‌ای می‌باشد که این به دلیل ماهیت این داده‌ساختار می‌باشد.

تقسیم فضا: هر عنصر داده‌ای در این ساختار یک کلید دارد که یک نقطه در فضای جستجوی چندبُعدی محسوب می‌شود. این فضا به n ناحیه تقسیم می‌شود که هر ناحیه متعلق به یک گره از شبکه است.

درخت افراز یک درخت دودویی است که با الحاق یک گره جدید به شبکه، یک ناحیه را به دو ناحیه جدید تقسیم می‌کند. ریشه درخت معرف کل فضای جستجو می‌باشد و هر گره میانی به‌طور بازگشتی فضا را به ناحیه‌های کوچک‌تر تقسیم می‌کند و این کار توسط معادلات صفحه انجام می‌شود. به عنوان مثال اگر گره v دارای دو فرزند راست و چپ r و l باشد، یکی در سمت راست صفحه‌ی v و دیگر در سمت چپ آن قرار می‌گیرد. همان‌طور که اشاره شد هر گره در شبکه متناظر با یک برگ در درخت افراز است و مسیری که ریشه را به آن گره متصل می‌کند، مسیر اصلی نامیده می‌شود. گره‌های میانی درخت شامل معادلات صفحه هستند که نشان می‌دهد گره مورد نظر در کدام طرف آن صفحه قرار دارد. به‌طور کلی مجموعه معادلات صفحه اختصاص یافته به یک مسیر اصلی را معادلات مشخصه آن می‌نامند که شامل اطلاعات مربوط به ناحیه‌ای است که آن گره در آن قرار دارد. هر گره در SkipTree علاوه بر معادله مشخصه خودش، معادلات مشخصه گره‌هایی که به آن‌ها متصل است را نیز نگهداری می‌کند. با توجه به این که ذخیره معادلات مشخصه در هر گره نیازمند $O(\log Nh)$ حافظه است ولی با عمل بهینگی حافظه که در این ساختار در نظر گرفته شده است، حافظه مورد نیاز به میزان قابل توجهی کاهش پیدا می‌کند. اگر ارتفاع درخت را h فرض کنیم، کل حافظه مورد نیاز در حالت متوسط برابر $O(\log N \log\log N)$ و در بدترین حالت برابر $O(\log^2 N)$ است.

اتصال‌ها در شبکه: به دلیل استفاده از درخت افراز یک ترتیب کلی میان گره‌ها برقرار است که با اتصال گره انتهایی به گره ابتدایی یک حلقه ایجاد می‌شود. اتصال مابین گره‌های درون حلقه در هر دوجهت می‌باشد ولی برای مسیریابی یک جهت که به مقصد نزدیک‌تر است انتخاب می‌شود و تا پایان مسیریابی از آن استفاده می‌شود. در حالت ایده‌آل هر

گره به تعداد $2\log_2 N - 1$ گره دیگر متصل است. هر گره به گره‌هایی با طول پرش 2^i در هر دوجهت متصل است که $i = 0 \cdots \log_2 N$ می‌باشد. همان طور که در قسمت‌های قبلی گفتیم برای اتصال بین گره‌ها از داده ساختار لیست پرشی استفاده می‌شود. لیست پرشی دوحالت دارد، یکی حالت ایده‌آل و دیگری حالت احتمالاتی. به دلیل این که در حالت ایده‌آل هزینه درج و حذف در آن زیاد است پس از حالت احتمالاتی آن استفاده می‌شود. در SkipTree هم از مدل احتمالاتی لیست پرشی استفاده شده است که هزینه آن کمتر است و بااحتمال بسیار زیاد عمل مسیریابی را در $O(\log_2 N)$ انجام می‌دهد. در این ساختار یک گره با احتمال $\frac{1}{2^h}$ دارای ارتفاع h است. ساختار SkipTree شامل حلقه‌هایی می‌شود که هر کدام در یک سطح قرار دارند. حلقه ریشه یا حلقه سطح صفر شامل تمام گره‌ها می‌شود. در سطح یک دو حلقه وجود دارد که حاصل تقسیم حلقه ریشه می‌باشد. هر حلقه در هر سطح به دو حلقه در سطح بالاتر تقسیم می‌شود و هر گره در هر حلقه به‌طور تصادفی به یکی از دو حلقه سطح بالاتر تعلق می‌گیرد که این کار از یک توزیع یکنواخت پیروی می‌کند. تعداد کل سطح‌ها هم برابر $O(\log_2 N)$ می‌باشد. در آخرین سطح هر گره به‌صورت منفرد یک حلقه تشکیل می‌دهد. به عنوان مثال یک حلقه در سطح iام شامل گره‌هایی می‌شود که به اندازه 2^i با هم فاصله دارند. در SkipTree یک شناسه عددی هم به هر گره تعلق می‌گیرد که هنگام الحاق یک گره به شبکه به آن اختصاص می‌گیرد. این شناسه یک عددی تصادفی بین شناسه عددی گره قبلی و گره بَعدی آن می‌باشد. این شناسه در جست‌وجوی بازه‌ای کاربر دارد.

۳ الگوریتم مسیریابی بهبودیافته

همان طور که در تشریح ساختار SkipTree و نحوه مسیریابی آن بیان شد، این ساختار عمل جستجو و مسیریابی را با احتمال بسیار زیاد در تعداد $O(\log_2 N)$ گام انجام می‌دهد. در این قسمت هدف ما بهبود کارایی عمل مسیریابی در SkipTree می‌باشد که باعث کاهش هزینه مسیریابی و کاهش تعداد اشاره‌گرها می‌شود. این الگوریتم با تغییراتی در نحوه ساخت و تکمیل جدول مسیریابی تعداد گام‌ها را به $O(\log_r D)$ کاهش می‌دهد. مقدار $D \ll N$ و r هم تعداد تقسیم هر حلقه در سطح بعد می‌باشد. به طور خلاصه ما با تغییر در تعداد تقسیمات حلقه‌ها و تغییر در تعداد گام‌ها، مسیریابی را بهبود می‌دهیم.

در ساختار SkipTree پایه هر حلقه در هر سطح به دو حلقه در سطح بالاتر تقسیم می‌شد که در ساختار SkipTree بهبودیافته هر حلقه به r (ring) حلقه در سطح بالاتر تقسیم می‌شود. به همین نحو هر گره به گره‌هایی با فاصله r^i متصل است یا به‌عبارت دیگر طول گام‌ها بلندتر خواهد بود. به عنوان نمونه فرض کنیم $r = 4$ و تعداد کل گره‌های شبکه هم $N = 1024$ باشد، حلقه ریشه در سطح صفر به چهار حلقه در سطح یک تقسیم می‌شود و باز هر حلقه در سطح یک به چهار حلقه دیگر در سطح دو تقسیم می‌شود و در کل ۱۶ حلقه در سطح دو داریم ($r^i = 4^2 = 16$). تعداد سطح‌ها در این حالت برابر ۵ ($\log_4 1024$) می‌باشد که در SkipTree پایه برابر ۱۰ ($\log_2 1024$) سطح می‌باشد، همین طور تعداد اشاره‌گرها و سطرهای جدول مسیریابی هم کاهش پیدا می‌کند و به مقدار $O(\log_r N)$ می‌رسد. به طورکلی با افزایش مقدار r، تعداد کل اشاره‌گرهای جدول مسیریابی کاهش می‌یابد، با کاهش تعداد اشاره‌گرها و افزایش فاصله میان گره‌هایی که به آن‌ها اشاره می‌شود بالطبع برای جستجوهای میانی باید هزینه‌ای پرداخت شود که باعث می‌شود تعداد گام‌های مسیریابی افزایش یافته و به تعداد $O(r \log_r N)$ برسد. همان‌طور که مشاهده می‌شود با ایجاد تغییر در SkipTree پایه و کاهش اندازه جدول مسیریابی، تعداد پرش‌ها افزایش می‌یابد و در کل هزینه مسیریابی نسبت به حالت پایه افزایش می‌یابد. حال برای رفع این مشکل یک جدول مسیریابی جدید پیشنهاد می‌کنیم که مسیریابی را با استفاده از همان اشاره‌گرهای بهبودیافته در تعداد گام کمتری انجام می‌دهد و ثابت می‌کنیم که با احتمال بسیار زیاد $O(\log_r N)$ خواهد بود. این جدول را جدول مسیریابی بهبودیافته یا به‌صورت مختصر ORT می‌نامیم.

۱.۳ نحوه ساخت جدول مسیریابی بهبودیافته

در این جدول جدید به ازاء هر سطر از جدول مسیریابی و در هر دو جهت، تعداد $r - 1$ اشاره‌گر جدید ذخیره می‌کنیم که این اشاره‌گرهای اضافی به گره‌های همسایه در آن سطح اشاره می‌کنند(گره‌های همسایه، گره‌هایی هستند که از لحاظ تعداد گام در "SkipTree Overlay" به گره مورد نظر نزدیک باشند). در این حالت تعداد پرش‌های مورد نیاز برای انجام عمل مسیریابی و جستجو کاهش می‌یابد و تعداد اشاره‌گرهای ذخیره شده در هر گره افزایش می‌یابد.

شبیه سازی‌های اولیه ما نشان می‌دهد که با انتخاب مقدار $r = 8$ یک توازن مناسب بین کارایی مسیریابی و تعداد اشاره‌گرهای ذخیره‌شده برقرار می‌شود.

۲.۳ خذف/اضافه شدن گره‌ها با الگوریتم جدید

زمانی که یک گره می‌خواهد با استفاده از جدول مسیریابی بهبودیافته به شبکه ملحق شود، یک پیام الحاق ایجاد می‌کند که این پیام باید در تمام حلقه‌هایی که گره جدید به آن‌ها تعلق دارد و در تمام سطح‌ها و همین طور در دوجهت حرکت کند و اطلاعات مربوط به حداقل $r - 1$ گره همسایه دیگر را جمع

آوری کند و این کار باید قبل از این‌که به حلقه بَعدی در سطح بعد برود انجام شود. هنگام انفصال هم کافی است گره مورد نظر قبل از ترک شبکه تمام همسایگانش را مطلع کند تا اتصال‌های خودشان را قطع کنند و جدول‌هایشان را به‌روز کنند.

۴ تحلیل الگوریتم بهبودیافته

در این قسمت ما به‌طور نظری ثابت می‌کنیم که با احتمال بسیار زیاد تعداد گام‌های مسیریابی در الگوریتم پیشنهادی برابر $O(\log_r D)$ است.به‌منظور سهولت در ادامه بحث تعاریف زیر را مطرح می‌کنیم:

- b : گره‌ای که عمل جستجو از آنجا آغاز می‌شود.
- e : گره‌ای که عمل جستجو در آنجا خاتمه می‌یابد.
- مسیر جستجو: دنباله‌ای از گره‌ها که در طول عمل مسیریابی ملاقات می‌شوند تا به مقصد برسیم.
- گام: هر مرحله در طول مسیر جستجو که از یک گره به گره بَعدی می‌رویم.
- لیست پرشی مشتق شده: حلقه‌هایی که b به آنها تعلق دارد یک ساختار لیست پرشی روی تمام گره‌ها ایجاد می‌کند که گره b در ابتدای لیست است.
- برای تحلیل مسیر جستجو در SkipTree، ما مسیری را در نظر می‌گیریم که الگوریتم جستجوی Skip List روی لیست پرشی مشتق شده استفاده می‌کند. سپس ثابت می‌کنیم که مسیر جستجوی SkipTree از مسیر جستجوی Skip List بیشتر نیست.
- T : مسیرجستجو در SkipTree از مبدأ تا مقصد
- L : مسیری که الگوریتم جستجوی Skip List در لیست پرشی مشتق شده استفاده می‌کند و از نقطه b شروع می‌شود.
- هر دو مسیر جستجو از b شروع شده و به e ختم می‌شوند و تمام گره‌ها در مسیر بین این دو قرار دارند.
- مسیرهای T و L لزوماً یکسان نیستند.

به‌منظور توصیف بهتر مسیرهای T و L از تابع زیر استفاده می‌کنیم،
$M(i,j)$: طولانی‌ترین پیشوند مشترک بین i و j (Match) در شناسه عددی. بر این اساس رابطه‌های زیر برقرار است:

$$M(i,j) = M(j,i) \quad (۱)$$

$$M(i,j) < M(j,k) \Rightarrow M(i,k) = M(i,j) \quad (۲)$$

$$M(i,j) \leq M(j,k) \Rightarrow M(i,k) \geq M(i,j) \quad (۳)$$

$$M(i,j) > u, M(i,k) > u \Rightarrow M(j,k) > u \quad (۴)$$

تعریف ۱ مسیر L شامل تمام گره‌های i بین b و e می‌شود به طوری که گره‌ای وجود ندارد که به e نزدیک‌تر باشد و پیشوند مشترک بیشتری با b داشته باشد، به‌عبارت دیگر L شامل $i \in [b,e]$ اگر و تنها اگر $\nexists j \in [i,e]$ به طوری که $M(b,j) > M(b,i)$.

تعریف ۲ مسیر T شامل تمام گره‌های i بین b و e می‌شود به طوری که گره‌ای وجود ندارد که به e نزدیک‌تر باشد و پیشوند مشترک بیشتری باگره قبلی‌اش داشته باشد. یعنی در مسیر T هرچه به مقصد نزدیک‌تر می‌شویم، پیشوند مشترک کمتری با قبلی‌ها داریم. این مسئله باعث می‌شود مسیر T به‌صورت یکتا تعریف شود. به عبارت دیگر $i \in [b,e]$ بلافاصله بعد از گره v در مسیر T می‌آید اگر و تنها اگر نزدیک‌ترین گره‌ای باشد که بعد از v می‌آید به طوری که $\nexists j \in [i,e]$ که $M(v,j) > M(v,i)$.

لم ۱ T یک زیر دنباله از L است، به این معنی که هر گره در T، در L هم وجود دارد.

اثبات: (فرض خلف)، فرض می‌کنیم که یک گره مثل i در مسیر T وجود دارد که در مسیر L نیست و i اولین گره در مسیر است که این ویژگی را دارد، یعنی قبلی‌ها همگی در دو مسیر مشترک بوده‌اند. واضح است که $i \neq b$ چون مبدأ در هر دو مسیر مشترک است. فرض می‌کنیم که v گره قبلی i در مسیر T می‌باشد. از آنجایی که $i \neq b$ پس i اولین گره در مسیر T نیست، پس v می‌تواند وجود داشته باشد. گره v به L هم تعلق دارد چون i اولین اختلاف محسوب می‌شود. سه حالت را در نظر می‌گیریم و در هر سه حالت به تناقض می‌رسیم.

حالت۱: فرض می‌کنیم که $M(b,i) > M(b,v)$. با توجه به تعریف ؟؟ چون $i \in [v,e]$ پس v نمی‌تواند در مسیر L باشد و این با فرض ما تناقض دارد.

حالت۲: فرض می‌کنیم که $M(b,i) = M(b,v)$. با توجه به تعریف ؟؟ $\nexists j \in [v,e]$ به طوری که $M(b,j) > M(b,v)$. چون $M(b,i) = M(b,v)$ و $[i,e] \subset [v,e]$ پس $\nexists j \in [i,e]$ به طوری که $M(b,j) > M(b,i)$ پس i در L وجود دارد که با فرض ما تناقض دارد.

حالت۳: فرض می‌کنیم که $M(b,i) < M(b,v)$. بر اساس رابطه ؟؟ داریم $M(b,i) = M(v,i)$. با توجه با این که i در L نیست، $\exists j \in [i,e]$ به طوری که $M(b,j) > M(b,i)$. با ترکیب $M(b,j) > M(b,v)$ و $M(b,v) > M(b,i)$ و رابطه

؟؟ نتیجه می‌شود $M(v,j) > M(b,i)$. از آن جایی که $M(b,i) = M(v,i)$ پس در $M(v,j) > M(v,i)$ صدق می‌کند و چون $j \in [i,e]$، پس با تعریف ؟؟، i در T نیست و با فرض ما تناقض دارد. □

نتیجه لم؟؟: طول مسیر جستجو در Skip List، طول مسیر جستجو در SkipTree را محدود می‌کند.

اگر D فاصله مبدأ تا مقصد باشد، برای Nهای بزرگ داریم: $D \ll N$

تعریف توابع توزیع احتمالاتی استاندارد:

تابع توزیع دوجمله‌ای:اگر هر آزمایش با احتمال $\frac{1}{r}$ موفق شود، $b(x;n,\frac{1}{r})$ برابر است با احتمال x بار موفقیت بعد از n آزمایش.

تابع توزیع تجمعی دوجمله‌ای: $B(x;n,\frac{1}{r})$ احتمال حداکثر x بار موفقیت بعد از n آزمایش.

تابع توزیع تجمعی دوجمله‌ای منفی: $B^*(n;x,\frac{1}{r})$ احتمال x بار موفقیت بعد از حداکثر n آزمایش.

بر این اساس روابط زیر برقرار هستند:

$$B^*(n;x,\tfrac{1}{r}) = 1 - B(x-1;n,\tfrac{1}{r}) \qquad (5)$$

$$B(\alpha n - 1;n,\tfrac{1}{r}) < \tfrac{1-\alpha}{1-\alpha r}\, b(\alpha n;n,\tfrac{1}{r}) \ for \ \ \alpha < \tfrac{1}{r} \qquad (6)$$

رابطه؟؟ به‌طور مستقیم با استفاده از تعاریف توابع توزیع تجمعی B و B^* نتیجه می‌شود و رابطه؟؟ هم با استفاده از قضیه C.4 صفحه ۱۱۱۹ مرجع [?] حاصل می‌شود که k باαn، P با $\frac{1}{r}$ و q با $1-\frac{1}{r}$ جایگزین شده است.

قضیه ۱ تعداد گام‌های مورد نیاز برای برای رسیدن به یک گره در فاصله D از مبدأ با استفاده از جدول مسیریابی در SkipTree پایه، برابر است با$O(r\log_r D)$. به‌عبارت دیگر مقدار $c_0 = 9$ وجود دارد به طوری که برای $c \geq c_0$ با احتمال حداقل $1-\frac{3}{\sqrt{e^c}}$، حداکثر تعداد گام برای جستجو برابر $(cr\log_r D + c^2 r)$ است.

اثبات: با استفاده از لم؟؟ کافی است که یک حد بالا برای تعداد گام‌ها در مسیر جستجوی Skp List پیدا کنیم. پس از این جا به بعد مسیر جستجو در Skip List را مد نظر قرار می‌دهیم. مقدار x را برابر $c+\log_r D$ تعریف می‌کنیم. متغیر تصادفی X برابر حداکثر سطح استفاده شده در مسیر جستجوی Skip List است. نشان می‌دهیم که $P[X \geq x]$ بسیار کم است. احتمال این که یک گره با گره مبدأ در x تا یا بیشتر پیشوند مشترک داشته باشند برابر است با $\frac{1}{r^x}$. (با توجه به تعریف Skip List احتمال این که یک گره ارتفاع x داشته باشد برابر $\frac{1}{r^x}$ است). در نتیجه، احتمال این که گره مبدأ با هر گره بین مبدأ و مقصد، در تعداد x یا بیشتر پیشوند مشترک باشد برابر است با $\frac{D}{r^x}$. بنابراین

$$P[X \geq x] \quad \leq \quad \frac{D}{r^x} = \frac{1}{r^{x-\log_r D}} = \frac{1}{r^c}$$

متغیر تصادفی Y را برابر تعداد گام‌های طی شده در مسیر جستجوی Skip List قرار می‌دهیم و f را برابر crx تعریف می‌کنیم. پس $f = (cr\log_r D + c^2 r)$. یک حد بالا برای احتمال این که Y بیشتر از f گام داشته باشد به دست می‌آوریم:

$$P[Y > f] = P[Y > f \ and \ X < x] + P[Y > f \ and \ X \geq x]$$
$$\leq P[Y > f \ and \ X < x] + P[X \geq x]$$

نشان می‌دهیم که احتمال این که جستجو بیشتر از f گام طول بشود بدون این که از یک اشاره‌گر سطح x عبور کند بسیار کم است. در Skip List برای تحلیل این مسئله و تعیین حد بالای احتمال از توزیع دوجمله‌ای منفی استفاده شده است [?] و نشان می‌دهد که $P[Y > f \ \ and \ \ X < x] \leq 1 - B^*(f;x,\frac{1}{r})$. با استفاده از رابطه؟؟ داریم $1 - B^*(f;x,\frac{1}{r}) = B(x-1;f,\frac{1}{r})$. قرارمی‌دهیم $\alpha = \frac{1}{cr}$ و بااعمال رابطه؟؟، حد بالای زیر حاصل می‌شود:

$$B(x-1;f,\frac{1}{r}) = B(\alpha f - 1;f,\frac{1}{r}) < \frac{1-\alpha}{1-\alpha r} b(\alpha f;f,\frac{1}{r})$$

از آن جایی که مقادیر c و r حداقل ۲ هستند، پس مقدار $\frac{1-\alpha}{1-\alpha r}$ حداکثر ۲ می‌شود. در نتیجه مقدار $B(x-1;f,\frac{1}{r})$ کوچک‌تر است از:

$$\begin{aligned}
& 2\binom{f}{x}(\frac{1}{r})^x(1-\frac{1}{r})^{f-x} \\
= \ & 2\binom{crx}{x}(\frac{1}{r})^x(1-\frac{1}{r})^{crx}(1-\frac{1}{r})^{-x} \\
< \ & 2\frac{(crx)^x}{x!}(\frac{1}{r})^x e^{-cx}(1-\frac{1}{r})^{-x} \\
< \ & 2e^{x\log crx}\left(\frac{1}{\sqrt{2\pi x}}\left(\frac{x}{e}\right)^{-x}\right)e^{-x\log r}e^{-cx}e^{x} \\
< \ & 2e^{x\log crx}e^{-x\log x}e^{x}e^{-x\log r}e^{-cx}e^{x} \\
\leq \ & 2e^{x(\log c+\log r+\log x)-x\log x+x-x\log r+x-cx} \\
= \ & 2e^{x\log c-cx} \\
= \ & 2e^{(-c+\log c+2)x}
\end{aligned}$$

در این اثبات این مسئله از تقریب‌های[?] استفاده شده است که همگی در ضمیمه ذکر شده‌اند. برای $c \geq 9$ داریم: $-c+\log c+2 < \frac{-c}{2} < 0$.بنابراین $e^{(-c+\log c+2)x} < e^{\frac{-c}{3}}$ در نتیجه $B(x-1;f,\frac{1}{r}) < 2e^{\frac{-c}{3}}$ با ترکیب نتایج به دست آمده داریم:

$$P[Y > f] \leq P[Y > f \ and \ X < x] + P[X \geq x]$$
$$\leq \frac{2}{e^{\frac{c}{3}}} + \frac{1}{r^c} < \frac{3}{\sqrt{e^c}}$$

با قرار دادن $c_0 = 9$ و برای $c \geq c_0$ داریم: $P[Y \leq f] \geq 1 - \frac{3}{\sqrt{e^c}}$ پس با احتمال بسیار زیاد و نزدیک به یک تعداد گام‌های ما از f کمتر است. □

لم ۲ حداکثر تعداد پرش‌های مورد نیاز در مسیر جستجوی T با استفاده از جدول مسیریابی بهبودیافته برابر است با $\frac{f}{r-1}+x+1$.

اثبات: فرض کنیم که $L=(b,z_1,\ldots,z_f)$، برابر با دنباله‌ی گره‌های مسیر L باشد به طوری که $z_f=e$ و $\forall i\geq 1\ ,M(b,z_i)\leq x$. بنابراین گره‌های z_i بر اساس مقدار $M(b,z_i)$ درون سطح‌ها افراز می‌شوند.(منظور levelهای لیست است). بر اساس لم؟؟ مسیر T هم شامل گره‌های z_i می‌شود و بر اساس تعریف جدول مسیریابی بهبودیافته این امکان را داریم که با یک پرش از تعداد $r-1$ گره در سطح $M(b,z_i)$ عبور کنیم. اگر فرض کنیم که تعداد n_i گره در هر سطح i از مسیر T داریم، در نتیجه T شامل حداکثر $\left\lceil\frac{n_i}{r-1}\right\rceil$ تا از آن گره‌ها می‌باشد و در نتیجه با جمع کردن تمام سطح‌ها، حداکثر تعداد گره‌ها برابر $\frac{f}{r-1}+x+1$ است.
می‌دانیم $\left\lceil\frac{x}{y}\right\rceil<\frac{x}{y}+1$ پس:

$$\sum_{i=0}^{x}\left\lceil\frac{n_i}{r-1}\right\rceil<\sum_{i=0}^{x}\left(\frac{n_i}{r-1}\right)+(x+1)=\frac{f}{r-1}+x+1$$

□

قضیه ۲ تعداد گام‌های مورد نیاز برای برای رسیدن به یک گره در فاصله D از مبدأ با استفاده از جدول مسیریابی بهبودیافته، برابر است با $O(\log_r D)$. به‌عبارت دیگر برای $c_0=9$ و $c\geq c_0$، عمل مسیریابی در حداکثر $(2c+1)\log_r D+2c^2+c+1$ گام با احتمال حداقل $1-\frac{3}{\sqrt{e^c}}$، انجام می‌شود.

اثبات: با استفاده از اثبات قضیه؟؟ و لم؟؟ و جایگزینی مقادیر داریم:

$$\begin{aligned}&\frac{f}{r-1}+x+1=\frac{crx}{r-1}+x+1\leq 2cx+x+1\\&=(2c+1)x+1=(2c+1)(c+\log_r D)+1\\&=(2c+1)\log_r D+2c^2+c+1\end{aligned}$$

□

۵ نتیجه

در این مقاله ما یک الگوریتم مسیریابی بهبودیافته معرفی کردیم که نسبت به الگوریتم‌های مسیریابی مشابه علاوه بر کارایی بهتر، از هزینه کمتری نیز برخوردار است. این الگوریتم با کاهش تعداد گام‌های مسیریابی از $O(\log_2 N)$ به $O(\log_r D)$ و همچنین کاهش اندازه جدول مسیریابی، بهبود قابل ملاحظه‌ای را در هزینه مسیریابی ایجاد می‌کند. در این‌جا D طول مسیر و r هم پارامتر بهبود مسیریابی می‌باشد. این الگوریتم بهبودیافته‌ی الگوریتم مسیریابی در داده‌ساختار SkipTree است که ما به‌صورت نظری میزان بهبود مسیریابی را اثبات کردیم. این الگوریتم علاوه بر امکان جست‌وجوی بازه‌ای، توانایی جست‌وجو روی داده‌های چندبُعدی را هم دارد و بدون تحمیل سربار خاصی به شبکه، هزینه مسیریابی را کاهش می‌دهد. این الگوریتم در مقایسه با الگوریتم‌های مسیریابی مشابه، به شکل درخت افراز و بُعد وابستگی ندارد و نیازی به متوازن بودن درخت و توزیع یکنواخت کلیدها نیست.

درآینده تصمیم داریم با پیاده‌سازی واقعی این الگوریتم و آزمون آن با داده‌های واقعی، کارایی آن را بهتر مورد بررسی و تجزیه و تحلیل قرار دهیم. همچنین قصد داریم با اضافه کردن تحمل‌پذیری خطا و امکان مسیریابی محلی آن را تقویت کنیم، چون با مسیریابی آگاه به توپولوژی می‌توان کارایی بهتری را در محیط واقعی ایجاد کرد.

ضمایم

تقریب‌های استفاده شده در اثبات مسائل،

$$\begin{aligned}\binom{crx}{x}&=\frac{(crx!)}{x!(crx-x)!}\qquad[1]-P1097\\&=\frac{crx(crx-1)\cdots(crx-x+1)}{x!}\leq\frac{(crx)^x}{x!}\end{aligned}$$

با استفاده از تقریب استرلینگ داریم:

$$\begin{aligned}x!&=\sqrt{2\pi x}\left(\frac{x}{e}\right)^x\left(1+\theta\left(\frac{1}{x}\right)\right)\ [1]-P55\ (3.17)\\\frac{1}{x!}&\leq\frac{1}{\sqrt{2\pi x}}\left(\frac{x}{e}\right)^{-x}\qquad x\geq 2\rightarrow\frac{1}{\sqrt{2\pi x}}<1\end{aligned}$$

تعریف تابع نمایی:

$$\lim_{r\to\infty}(1-\frac{1}{r})^r=e^{-1},(1-\frac{1}{r})^r<e^{-1},(1-\frac{1}{r})^{crx}<e^{-cx}$$

مراجع

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* 2nd ed. 2001.

[2] S. Alaei, M. Toossi,and M. Ghodsi, *Skip-Tree: A Scalable Range-Queryable Distributed Data Structure for Multidimensional Data,* (ISAAC'2005).

[3] W. Pugh. *A Skip List Cookbook.* Technical Report, University of Maryland, 1990.

[4] W. Pugh. *Skip Lists: A Probabilistic Alternative to Balanced Trees.* In Workshop on Algorithms and Data Structures, 1989.