سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران
جزیره کیش، خلیج فارس، ایران
۱۹ الی ۲۱ اسفند ۱۳۸۶
مقاله کامل
انجمن کامپیوتر ایران
Computer Society of Iran
دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

مسیریابی آگاه به هم‌بندی در داده‌ساختارهای توزیع‌شده‌ی مبتنی بر لیست‌های پرشی

جواد شاهپریان
کارشناس ارشد
دانشکده‌ی مهندسی کامپیوتر
دانشگاه صنعتی شریف
shahparian@ce.sharif.edu

سید ایمان میررضایی
کارشناس ارشد
دانشکده‌ی مهندسی کامپیوتر
دانشگاه صنعتی شریف
mirrezaei@ce.sharif.edu

محمد قدسی [۱]
استاد دانشکده‌ی مهندسی کامپیوتر
دانشگاه صنعتی شریف
و مرکز تحقیقات فیزیک نظری و ریاضیات
ghodsi@sharif.edu

چکیده: در این مقاله ما به ارائه یک الگوریتم جدید و کارآمد برای مسیریابی آگاه به هم‌بندی فیزیکی در شبکه‌های P2P می‌پردازیم. این الگوریتم با در نظر گرفتن همسایگی در لایه فیزیکی، عمل مسیریابی پیام‌ها را با توجه به هم‌بندی لایه فیزیکی شبکه انجام می‌دهد. این ویژگی به داده‌ساختار SkipTree بهبود یافته اضافه شده است. SkipTree بهبود یافته از مسیریابی آگاه به هم‌بندی پشتیبانی نمی‌کرد و انتخاب گره‌های همسایه بدون توجه به هم‌بندی فیزیکی صورت می‌گرفت.

ما در این مقاله با اضافه کردن جدول همسایگی و ارائه یک الگوریتم مسیریابی آگاه به هم‌بندی تمام ویژگی‌های داده‌ساختار قبلی را حفظ می‌کنیم و مسیریابی را با توجه به همسایگیِ گره‌ها در لایه فیزیکی انجام می‌دهیم. نتایج ارزیابی و پیاده‌سازی شبکه رویه‌ای جدید هم بهبود کارایی را در این مدل تأیید می‌کند. این شبکه رویه‌ای جدید را «SkipTree بهبود یافته‌ی آگاه به هم‌بندی» می‌نامیم که علاوه بر کاهش هزینه تاخیر نسبی در شبکه، باعث ایجاد محلیت مسیر، محلیت محتوا و مزایای امنیتی هم می‌شود.

واژه‌های کلیدی
شبکه‌های P2P، مسیریابی آگاه به هم‌بندی، جدول همسایگی، لیست پرشی، داده‌ساختارهای توزیع‌شده

۱ مقدمه

یک شبکه رویه‌ایِ P2P یک شبکه منطقی است که روی یک شبکه فیزیکی ایجاد می‌شود. شبکه رویه اجزاء یک شبکه را به‌صورت منطقی سازماندهی می‌کند و هر گره از طریق همسایگانش به شبکه رویه‌ای متصل است. در بسیاری از شبکه‌های رویه‌ای موجود از قبیل CAN [۱۹]، Chord [۱۱]، Pastry [۱۰]، KazaA [۱۷] و Gnutella [۱۸]، وقتی که یک گره جدید به شبکه ملحق می شود بدون توجه به هم‌بندی فیزیکی و شبکه زیرین مکان‌یابی می‌شود، که عدم توجه به این مسئله باعث می‌شود که کارایی شبکه کاهش پیدا کرده و مقدار زیادی ترافیک اضافی به اتصالات فیزیکی شبکه تحمیل شود. انتخاب همسایه‌ها در این نوع از شبکه‌ها بدون آگاهی از شبکه فیزیکی زیرین و به‌صورت تصادفی بوده که این امر باعث عدم هماهنگی بین لایه‌ی رویه و لایه‌ی فیزیکی می‌شود. RAQ [۸] از جمله داده‌ساختارهایی است که مسئله‌ی آگاهی از هم‌بندی را پشتیبانی نمی‌کند. ما با تغییراتی که در آن اعمال کردیم، یک شبکه‌ی رویه‌ای جدید به‌نام RAQNet [۳] ارائه کردیم که در آن ویژگی مسیریابی آگاه به هم‌بندی به داده‌ساختار RAQ اضافه‌شده است.

به منظور ایجاد ویژگی مسیریابی آگاه به هم‌بندی و توجه به مجاورت در داده‌ساختار SkipTree، ما از یک جدول مسیریابی اضافی استفاده کردیم. در این شبکه رویه‌ای جدید ما یک جدول مسیریابی دیگر به نام جدول همسایگی معرفی می‌کنیم که مسئله‌ی مطرح شده‌ی مربوط به مجاورت در شبکه را رفع می‌کند. هدف از ایجاد جدول همسایگی این است که بتوانیم عمل مسیریابی را در تعداد $O(\log N)$ گام انجام دهیم البته با توجه به این که هر گام دارای کمترین هزینه از لحاظ تأخیر شبکه باشد.

در این مقاله ما با اضافه کردن یک جدول مسیریابی اضافی سعی کردیم تأخیر جستجو و مسیریابی را تا حد

[۱] این نویسنده توسط مرکز تحقیقات فیزیک نظری(IPM) (قرارداد شماره CS2386-2-01) حمایت شده است.

روش محلیت محتوایی و محلیت مسیر می‌باشد، به این معنی که داده‌هایی که از محتوای مرتبط به هم برخوردارند و مربوط به یک ناحیه خاص می‌شوند تقریباً در یک ناحیه ذخیره می‌شوند و به همین صورت پیام‌هایی هم که مربوط به یک ناحیه خاص هستند از مسیر محلی آن ناحیه خارج نمی‌شوند و محلیت مسیر هم حفظ می‌شود. این کار از مزایای امنیتی هم برخوردار است و می‌تواند با ایجاد نواحی جداگانه، امنیت هر منطقه را تضمین کند. در این حالت داده‌های یک سازمان در سازمان دیگری ذخیره نمی‌شود و همین طور پیام‌های مربوط به یک سازمان از مسیر سازمان دیگری عبور نمی‌کند و با حفظ محلیت محتوایی و محلیت مسیر، امنیت هم ایجاد می‌شود. به طور کلی توجه به مجاورت و هم‌بندی فیزیکی در شبکه یکی از مسائلی است که امروزه مورد توجه زیادی قرارگرفته است.

در ادامه در بخش ۲ به مروری بر ساختار SkipTreeپایه و بهبودیافته می‌پردازیم، در بخش ۳ به بیان مجاورت در شبکه و ایجاد جدول همسایگی پیشنهادی می‌پردازیم، در بخش ۴ به ارائه نتایج تجربی پرداخته و در پایان هم در بخش ۵ به بیان نتیجه کارهای انجام شده می‌پردازیم.

۲ مروری برSkipTreeپایه و بهبودیافته

SkipTree یک داده‌ساختار توزیع‌شده‌ی متوازن در شبکه‌های P2P است که به منظور 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. 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 اتصالات میان گره‌ها که از روشی شبیه ساختار لیست پرشی استفاده می‌شود.

ما در [۲] با ایجاد تغییراتی در نحوه ایجاد اشاده‌گرها و اتصالات میان گره‌ها و همچنین ایجاد تغییراتی در نحوه ساخت جداول مسیریابی در SkipTree [۵] مسیریابی آن را بهبود دادیم. به دلیل این که در این داده‌ساختار از لیست‌های پرشی برای مسیریابی استفاده شده می‌توان با تغییر تعداد پرش‌ها و تغییر در نحوه ایجاد اشاره‌گرها بهبود قابل ملاحظه‌ای را در مسیریابی آن ایجاد کرد، به طوری که می‌یابد که در ان D طول مسیر و r فاکتور میزان بهبود مسیریابی در شبکه است. به طور کلی هزینه مسیریابی از لحاظ تعداد گام‌های مورد نیاز و همچنین اندازه جدول مسیریابی و حافظه مورد نیاز آن کاهش پیدا کرده است. در این مقاله هم با تکیه بر تجربیاتی که در [۱] حاصل شده مسیریابی آگاه به هم‌بندی را در این داده‌ساختار اعمال کرده‌ایم.

۱.۲ الگوریتم بهبود مسیریابی در SkipTree

همان طور که در تشریح ساختار SkipTree [۵] 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. ما با ایجاد تغییراتی در نحوه ساخت و تکمیل جدول مسریابی، تعداد گام‌ها را به $O(\log_r D)$ کاهش دادیم. 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 دادیم.

در ساختار SkipTree پایه [۵] هر حلقه در هر سطح به دو حلقه در سطح بالاتر تقسیم می‌شد که در ساختار SkipTree بهبودیافته [۲] 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 به عبارت دیگر طول گام‌ها بلندتر خواهد بود. به عنوان نمونه فرض کنیم $r = ۴$ و تعداد کل گره‌های شبکه هم $N = ۱۰۲۴$ 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 ($r^i = ۴^۲ = ۱۶$). تعداد سطوح در این حالت برابر ۵ ($\log_4 ۱۰۲۴$) می‌باشد که در SkipTree پایه برابر ۱۰ ($\log_2 ۱۰۲۴$) سطح می‌باشد، همین طور تعداد اشاره‌گرها و سطرهای جدول مسیریابی هم کاهش پیدا می‌کند و به مقدار $O(\log_r N)$ می‌رسد. به طورکلی به 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

حال برای رفع این مشکل یک جدول مسیریابی جدید پیشنهاد کرده‌ایم که مسیریابی را با استفاده از همان اشاره‌گرهای بهبود یافته در تعداد گام کمتری انجام می‌دهد و ثابت کرده‌ایم که با احتمال بسیار زیاد $O(\log_r N)$ خواهد بود[۲]. این جدول را جدول مسیریابی بهبود یافته می‌نامیم.

۳ توجه به مجاورت در شبکه و ایجاد جدول همسایگی

در مدل احتمالاتی که برای داده‌ساختار لیست پرشی حلقوی معرفی شد، همسایه‌های یک گره به صورت تصادفی تعیین می‌شوند زیرا عضویت گره‌ها در حلقه‌ها به صورت تصادفی انجام می‌شود و در نتیجه همسایگان هر گره هم تصادفی انتخاب می‌شوند. البته اشاره به این نکته ضروری است که گره‌های هر حلقه بر حسب شناسه اسمی مرتب هستند.

ساختار SkipTree پایه بدون توجه مستقیم به توپولوژی فیزیکی شبکه طراحی شده و این مسئله باعث لطمه خوردن به کارایی مسیریابی می‌شود. به عنوان مثال فرض کنید که می‌خواهیم یک پیام را از گره A در تهران به گره B در کرج بفرستیم و ممکن است که این پیام از طریق یک گره میانی در شیراز مسیریابی شود. این مسئله باعث می‌شود که پیام مسیر بیشتری را طی کند، در صورتی که اگر از یک گره میانی در دیگر در استان تهران عبور می‌کرد مسیر کوتاه‌تری را می‌پیمود.

به منظور رفع این مشکل می‌توان از یک جدول مسیریابی اضافی استفاده کرد. در این جا ما یک جدول مسیریابی دیگر به نام جدول همسایگی معرفی می‌کنیم که مسئله مطرح شده را تا حدود زیادی رفع می‌کند. هدف از ایجاد جدول همسایگی این است که بتوانیم عمل مسیریابی را در تعداد $O(\log N)$ گام انجام دهیم البته با توجه به این که هر گام دارای کمترین هزینه از لحاظ تأخیر شبکه‌باشد. جدول همسایگی که ما در اینجا معرفی کردیم تا حدودی از جدول مسیریابی آگاه به مجاورت در Pastry [۱۳] الهام گرفته شده است. به منظور ایجاد ویژگی همسایگی شبکه‌ای در داده‌ساختار SkipTree، می‌توان به این صورت عمل کرد که هر گره که تقریباً از لحاظ شناسه اسمی نزدیکی کافی به گره مورد نظر دارد به عنوان یکی از مدخل‌های جدول مسیریابی قرار بگیرد. به عنوان مثال لازم نیست که یک مدخل از جدول همسایگی در سطح سوم حتماً به گره‌ای جای ان به گره‌هایی اشاره کند که هفت و یا نه گره با ان فاصله دارند. در این شرایط به طور آماری تعداد گام‌هایی که برای مسیریابی لازم است تا یک پیام به مقصد برسد مشابه همان حالت اولیه است. اگرچه این نکته را هم باید در نظر گرفت که اگر هفتمین گره و یا نهمین گره از لحاظ فاصله‌ی شبکه‌ای نزدیک‌تر باشند، در کل میزان کارایی مسیریابی نسبت به حالتی که از گره هشتم استفاده می‌شد بهتر است. در حقیقت مدخل جدول همسایگی در سطح h می‌تواند هرجایی بین گره‌های ۲^h و $۲^{h+۱}$ باشد و کارایی مسیریابی هم در حد $O(\log N)$ باقی خواهد ماند. به عبارت دیگر گره‌ای که می‌خواهد برای مدخل جدول همسایگی در سطح h انتخاب شود می‌تواند در بازه‌ی ۲^h تا $۲^{h+۱}$ باشد و این بازه هم شامل گره‌های قبل از گره مورد نظر می‌شود و هم بعد از آن.

به منظور ساخت جدول همسایگی، یک گره باید مجموعه‌ای از گره‌های همسایه را که از لحاظ فاصله‌ی شبکه‌ای نزیک به آن هستند تعیین کند. بر خلاف Chord و Pastry نمی‌توان به راحتی با نگاه کردن به شناسه اسمیِ گره‌های کاندید از طریق حلقه ریشه فاصله‌ها را تخمین زد. برای حل این مشکل از جدول مسیریابی اصلی گره استفاده می‌کنیم. جدول مسیریابی حلقه ریشه را به حلقه‌هایی تقسیم می‌کند که فواصل آنها به طور نمایی افزایش پیدا می‌کند. بنابراین دو اشاره‌گر در سطوح مجاور جدول همسایگی یک بازه از شناسه اسمی در حلقه ریشه ایجاد می‌کنند. به منظور شناسایی گره‌های کاندیدی که به گره مورد نظر نزدیک هستند از فرآیند بازگشتی زیر استفاده می‌کنیم: ما از یک گره نزدیک به نام گره بَذر (Seed Node) شروع می‌کنیم و بقیه‌ی گره‌های نزدیک را با پرس‌وجو از جدول همسایگی گره بذر پیدا می‌کنیم. در نهایت دو گره را تعیین می‌کنیم که به یگدیگر نزدیک هستند و این کار نیز با استفاده از تخمین میزان تأخیر رفت و برگشت محاسبه می‌شود. در بخش بعدی الگوریتمی که گره‌ها برای ایجاد و ساخت جدول همسایگی استفاده می‌کنند به طور مفصل و جزئی توضیح داده خواهد شد. بعد از این که جدول همسایگی اولیه ساخته شد، سعی می‌شود که به طور دائمی کیفیت مدخل‌های جدول همسایگی و اشاره‌گرهای آن بهبود یابد به این معنی که چون دائماً گره‌های جدیدی به شبکه الحاق می‌شوند و یا شبکه را ترک می‌کنند پس لازم است همان طور که اشاره‌گرهای جدول مسیریابی اصلی به‌روز می‌شوند، اشاره‌گرهای جدول همسایگی هم به‌روز شوند و در صورتی که کاندید بهتری برای مدخل‌های

هم توسط یک الگوریتم پایدارکننده(پایاساز) انجام می‌شود که این الگوریتم به طور دوره‌ای روی گره‌های شبکه اجرا می‌شود. این الگوریتم دوره‌ای پایاساز بسیار شبیه به الگوریتمی است که برای ساخت اولیه جدول همسایگی استفاده می‌شود و در بخش بعد توضیح داده شده است.

۱.۳ نحوه ساخت جدول همسایگی

همان طور که اشاره شد جدول مسیریابی اصلی دارای یک پارامتر r است که تعداد حلقه‌های مورد استفاده برای تقسیم را مشخص می‌کند. جدول همسایگی هم این پارامتر را از جدول مسیریابی ارث می‌برد چون در حقیقت جدول همسایگی بر اساس جدول مسیریابی ایجاد می‌شود. البته در برخی موارد این امکان وجود دارد که یک جدول همسایگی را با پارامترهای متفاوتی نسبت به جدول مسیریابی ایجاد کنیم و این کار به این صورت انجام می‌شود که ابتدا یک جدول مسیریابی موقت با استفاده از پارامترهای مورد نظر می‌سازیم و پس از ایجاد جدول همسایگی از روی آن، جدول همسایگی موقت را حذف می‌کنیم. به عنوان مثال فرض کنید که جدول مسیریابی ما دارای پارامتر $r = ۲$ است و ما می‌خواهیم یک جدول همسایگی با پارامتر $r = ۸$ بسازیم، برای این کار ابتدا یک جدول مسیریابی موقت با پارامتر $r = ۸$ می‌سازیم که به عنوان ورودی الگوریتم ساخت جدول همسایگی استفاده می‌شود و از روی آن جدول همسایگی مورد نظر را ایجاد می‌کنیم و بعد از آن جدول مسیریابی موقت را حذف می‌کنیم.

زمانی که یک گره جدید به شبکه ملحق می‌شود، ابتدا جدول مسیریابی خود را می‌سازد. سپس برای ساخت جدول همسایگی به این صورت عمل می‌شود که اول تمام سطرها یا مدخل‌های جدول مسیریابی گره مورد نظر درون یک لیست جدا کپی می‌شود که این لیست بر اساس شناسه اسمی مرتب می‌شود و پس از آن سطرهای تکراری حذف می‌شوند. به علت ماهیت احتمالاتی که در ساخت جدول مسیریابی وجود دارد سطرهای تکراری و خارج از ترتیب در این لیست وجود خواهند داشت.

پس از ساخت جدول همسایگی توسط گره جدید ملحق شده به شبکه، این گره اقدام به ایجاد یک پیام «الحاق به جدول همسایگی» می‌کند. این پیام حاوی همان لیست مرتب شده‌ی اشاره‌گرها می‌باشد.یک لیست که حاوی i گره است دقیقاً شامل $i-۱$ بُرهه می‌باشد،(منظور فاصله میان گره‌ها است). گره جدیدی که به شبکه الحاق گره بذر می‌فرستد. گره بذر گره‌ای است که باید از لحاظ فاصله‌ی شبکه‌ای نزدیک به گره مورد نظر است.

هر گره‌ای که یک پیام الحاق به جدول همسایگی را دریافت می‌کند با استفاده از سطرهای جدول همسایگی خودش اقدام به پرکردن بُرهه‌های لیست مورد نظر می‌کند و این کار را با استفاده از گره‌های کاندید انجام می‌دهد. به منظور جلوگیری از انباشتگی و تراکم زیاد گره‌های کاندید و همچنین ملاحظات عملی در پیاده‌سازی، حداکثر تعداد گره‌های کاندید به ازاء هر بُرهه را به تعداد ۱۰ گره محدود می‌کنیم. بعد از پرکردن بُرهه‌هایی که امکان پذیر هستند، گره بررسی می‌کند که آیا هنوز بُرهه‌ای وجود دارد که خالی مانده باشد. اگر چنین بود و هنوز بُرهه یا بُرهه‌هایی وجود داشت که برای آن گره کاندید وجود نداشت، این گره باید پیام الحاق به جدول همسایگی را به گره دیگری ارسال کند تا بُرهه‌هایی که خالی مانده‌اند پرشوند.

فرض کنید که در ابتدا تمام بُرهه‌ها خالی هستند، تعداد گام‌هایی که انتظار می‌رود برای پیدا کردن یک گره کاندید برای iامین دورترین بُرهه از گره الحاقی(گره‌ای که می‌خواهد به شبکه الحاق شود) نیاز باشد برابر است با $O(i)$. به عبارت دیگر اگر بخواهیم یک کاندید برای بُرهه‌ای به فاصله i از گره الحاقی پیدا کنیم، معمولاً نیاز به $O(i)$ گام است. در نتیجه به منظور پیدا کردن کاندیدهایی که از لحاظ مجاورت شبکه‌ای به گره الحاقی نزدیک باشند از استراتژی زیر استفاده می‌کنیم: گره‌هایی که پیام الحاق را دریافت می‌کنند با استفاده از سطرهای جدول همسایگی خودشان پیام را به بُرهه‌هایی که خالی هستند و نسبت به گره الحاقی دورترین هستند ارسال می‌کنند. اگر تمام بُرهه‌ها حداقل یک کاندید داشتند، گره‌ای که در حال حاضر پیام را در اختیار دارد پیام تکمیل الحاق را به گره الحاقی اصلی باز می‌فرستد. تعداد کل گام‌های مورد انتظار برای پرکردن تمام بُرهه‌ها برابر $O(\log N)$ است.

زمانی که گره اصلی پیام الحاق خودش را دریافت می‌کند، یک گره کاندید از میان گره‌های کاندید به ازاء هر بُرهه انتخاب می‌کند و آن را به عنوان مدخل یا سطر جدول همسایگی‌اش قرارمی‌دهد. انتخاب یک گره از میان کاندیدهای موجود نیز بر اساس تخمین میزان تأخیر شبکه است و هر گره‌ای که کمترین تأخیر را با گره مورد نظر داشت به عنوان سطر جدول همسایگی انتخاب می‌شود.

ما در این جا جزئیات کلیدی ساخت جدول همسایگی را که باقی مانده است توضیح می‌دهیم. از آنجایی که امکان

دارد، جدول همسایگی شامل بُرهه‌هایی است که فضای آدرس‌دهی را برای گره الحاقی در هر دو جهت پوشش می‌دهد. بنابراین گره الحاقی دو عدد پیام الحاق ایجاد می‌کند و در هر دو جهت ارسال می‌کند.

میزان کارآمدی جدول همسایگی وابستگی بسیار زیادی به پیداکردن گره‌های نزدیک دارد. پایه و اساس این فرآیند پیداکردن یک گره بذر مناسب است. به طور کلی شبکه‌های رویه‌ای مشابهی که اقدام به ساخت جدول همسایگی می‌کنند معمولا برای تعیین گره بذر یکی از دو استراتژی زیر را در شبیه‌سازهای خود به کار می‌گیرند. در استراتژی اول اطلاعات جامعی که از توپولوژی شبیه‌ساز وجود دارد استفاده می‌شود تا بتوانند نزدیک‌ترین گره را تعیین کنند. در این استراتژی با استفاده از اطلاعات کاملی که از کل توپولوژی وجود دارد گره‌های نزدیک انتخاب می‌شوند. در استراتژی دوم که به واقعیت هم نزدیک‌تر است گره بذر به‌صورت تصادفی انتخاب می‌شود و سپس الگوریتم الحاق به جدول همسایگی دو بار اجرا می‌شود. اجرای اول الگوریتم الحاق به جدول همسایگی برای تعیین یک گره بذر مناسب است و اجرای دوم برای ساختن یک جدول همسایگی بهتر از روی گره بذر پیدا شده است. به عبارت دیگر ابتدا یک گره بذر به طور تصادفی انتخاب می‌شود و در اجرای اول الگوریتم یک گره بذر مناسب پیدا می‌شود. دراجرای دوم الگوریتم هم جدول همسایگی بهتر می‌شود. به منظور پیاده‌سازی کار فرضیات زیر را در نظر می‌گیریم: برای انتخاب گره بذر، فاصله‌ی گره را با تعدادی از همسایگان چپ و راستش که از لحاظ شناسه اسمی به آن نزدیک‌ترین هستند محاسبه می‌کنیم. معمولاً این تعداد این گره‌ها ۱۶ عدد می‌باشد که ۸ عدد در چپ و ۸ عدد در راست قراردارد. فاصله گره از لحاظ تأخیر شبکه با تمام این ۱۶ همسایه محاسبه می‌شود و نزدیک‌ترین گره به عنوان گره بذر انتخاب می‌شود. بعد از این که جدول همسایگی اولیه ساخته‌شد سعی می‌شود که به‌طور مرتب کیفیت سطرهای این جدول بهبود یابد و متناسب با الحاق و حذف گره‌ها تنظیم شود و این کار به‌وسیله‌ی یک الگوریتم پایاساز انجام می‌شود که به طور دوره‌ای اجرا خواهدشد. جدول همسایگی به صورت دوره‌ای و در فواصل معین به‌روز می‌شود در نتیجه اشاره‌گرهای جدول همسایگی می‌توانند به درستی نحوه توزیع شناسه‌های اسمی را نشان دهند. مکانیزم دوره‌ای که برای به‌روزرسانی جداول همسایگی استفاده می‌شود بسیار شبیه به الگوریتم ساخت اولیه‌ی جدول همسایگی است که در بالا توضیح داده‌شد ولی در مکانیزم به‌روزرسانی و ساخت جدول همسایگی اولیه این است که برای به روزرسانی، علاوه بر گره‌های کاندیدی که پیام الحاق جدول همسایگی را بر می‌گرداند، سطرهای جدول همسایگی فعلی هم به عنوان گره‌های کاندید معرفی می‌شوند. تفاوت دیگری که وجود دارد این است که برای به‌روزرسانی جدول گره‌ی بذر به عنوان بهترین کاندید از جدول همسایگی موجود انتخاب می‌شود. تفاوت آخر هم این که سطرهای جدول همسایگی ممکن است با الحاق و یا حذف گره‌های جدید به‌روز رسانی شوند. به عبارت دیگر ممکن است که با یک پیام شبکه‌ای، یک جدول همسایگی به‌روزرسانی شود و این به‌روزرسانی خارج از دوره‌های به‌روزرسانی باشد.

۴ نتایج تجربی و ارزیابی کارایی

به منظور فهم و ارزیابی طراحی و کارایی داده‌ساختار SkipTree بهبودیافته، ما از یک نوع پیاده‌سازی که در سطح بسته‌های(Packets) ساده‌ی شبکه استفاده کرده‌ایم که تعداد بسته‌های ارسال شده روی یک اتصال فیزیکی را می‌شمارد و به این اتصال یک گام و یا یک تأخیر خاص اختصاص می‌دهد که این موضوع بستگی به نوع توپولوژی مورد استفاده در شبکه دارد. در این جا ما تأخیر صف و یا بسته‌های از بین رفته را مدل نمی‌کنیم به علت این که مدل کردن این موارد شبیه‌سازی تعداد زیاد گره و شبیه‌سازی شبکه‌های بزرگ را بسیار دشوار می‌کند. برنامه ما دو نوع شبکه‌ی رویه‌ای را پیاده‌سازی می‌کند، یکی شبکه‌ی رویه‌ای SkipTree پایه و دیگری شبکه‌ی رویه‌ای SkipTree بهبودیافته. در پیاده‌سازی این شبیه‌ساز از پیاده‌سازی نمونه Pastry [۱۰] الهام گرفته‌شده و در برخی موارد از ایده‌های آن ها در طراحی لایه‌ی شبکه استفاده‌شده است. توپولوژی یا هم‌بندی مورد استفاده در این ارزیابی، توپولوژی GT-ITM [۱۴] بوده که توپولوژی GT-ITM با استفاده از تولید کننده‌ی تصادفی گراف Georgia Tech ایجاد می‌شود.

به‌طور کلی کد برنامه SkipTree بهبودیافته می‌تواند در یکی از دو محیط زیر اجرا شود:

- محیط شبیه‌سازی: در این محیط تمام گره‌ها در داخل یک برنامه اجرا می‌شوند و از طریق یک توپولوژی شبکه‌ی شبیه‌سازی شده با یکدیگر ارتباط برقرار می‌کنند.

عنوان یک برنامه جدا و مستقل اجرا می‌شود و از طریق اتصالات TCP با گره‌های دیگر ارتباط برقرار می‌کند. محیط خوشه‌ای یک محیط واقعی محسوب می‌شود و دقیقاً رفتار یک شبکه واقعی را دارد.

محیط شبیه‌سازی این امکان را به ما می‌دهد که بتوانیم تجربیات خود را با تعداد زیادی گره آزمایش کنیم در صورتی که این امکان در محیط خوشه‌ای یا همان محیط واقعی وجود ندارد. البته محیط خوشه‌ای هم این امکان را می‌دهد که تجربیات خود را به صورت واقعی آزمایش کنیم.

اشاره به این نکته در مورد محیط خوشه‌ای ضروری است که همان طور که این امکان وجود دارد که هر گره به عنوان یک برنامه جدا در این محیط کار کند، این امکان هم وجود دارد که چند گره به همراه چند برنامه روی یک ماشین فیزیکی اجرا شوند. این کار به این صورت انجام می‌شود که هر گره به یک پورت خاص گوش می‌دهد و در کل هر گره از لحاظ پورت متفاوت است. به عنوان مثال می‌توان برای داشتن تعداد ۱۰۰ عدد گره از ۱۰ عدد ماشین استفاده کرد که روی هر کدام ۱۰ عدد گره اجرا شده است.

۱.۴ معیارهای ارزیابی

معیارهای ما برای ارزیابی کارایی شبکه شامل این موارد است:

- **هزینه‌ی تأخیر نسبی** (RDP:Relative Delay Penalty): نسبت تأخیر در مسیر بین دو گره در شبکه‌ی رویه‌ای نسبت به تأخیر مسیر در لایه‌ی زیرین.
- **فاصله‌ی فیزیکی شبکه:** طول مسیر بین دو گره در شبکه‌ی رویه‌ای برحسب فاصله در شبکه‌ی زیرین. به این معنی که فاصله را برحسب مقادیر واقعی در شبکه زیرین اندازه‌گیری کنیم. فاصله را می‌توان برحسب تعداد گام و یا برحسب میزان تأخیر شبکه محاسبه کرد.

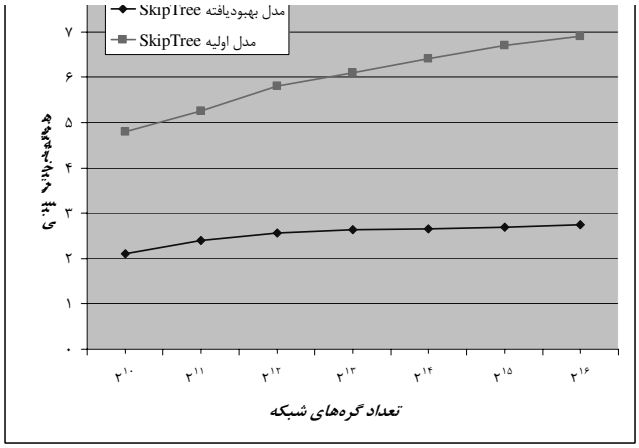
تعاریف مورد استفاده در آزمایشات:

- گام‌های برنامه‌کاربردی: تعداد گام‌هایی که در سطح برنامه کاربردی نیاز است تا یک پیام از طریق شبکه‌ی رویه‌ای به سمت مقصد مسیریابی شود. لایه کاربرد یا لایه‌ی برنامه کاربردی، لایه‌ی "Application Layer" روی "Overlay Layer" قرار دارد. معمولاً در سطح لایه‌ی کاربرد، برنامه‌های کاربردی مثل برنامه‌های ارسال پیام کار می‌کنند.
- هزینه‌ی تأخیر نسبی: نسبت میانگین تأخیر مسیریابی با استفاده از شبکه‌ی رویه‌ای به میانگین تأخیر مسیریابی با استفاده از لایه‌ی زیرین.
- نوع شبکه‌ی رویه‌ای: SkipTree پایه و SkipTree بهبودیافته
- توپولوژی: GT-ITM
- گره‌ها(N): تعداد گره‌های شبکه‌ی رویه‌ای. تعداد گره‌های شرکت کننده در هر آزمایش از ۲۸ تا ۲۱۶ است.
- تعداد دفعات آزمایش: معمولاً هر آزمایش ۱۰ بار با گره‌های بذر متفاوت تکرار می‌شود و میانگین آن‌ها درج می‌شود.

۲.۴ هزینه‌های مسیریابی

به منظور ارزیابی کارایی مسیریابی در داده‌ساختار SkipTree ما تعداد ۱۰۲۴(۲۱۰) گره تا ۶۵۵۳۶(۲۱۶) گره را در شبکه‌ی رویه‌ای مورد بررسی قراردادیم. یکی از اهداف این کار بررسی هزینه‌ی تأخیر نسبی در دو شبکه‌ی رویه‌ای ارائه شده است.

شکل ۱ هزینه‌ی تأخیر نسبی را در دو مدل SkipTree پایه و SkipTree بهبودیافته نشان می‌دهد. همان طور که مشخص است معیار مقایسه‌ی این دو شبکه هزینه‌ی تأخیر نسبی می‌باشد که با تعداد ۱۰۲۴(۲۱۰) گره تا ۶۵۵۳۶(۲۱۶) گره این آزمایش انجام شده و نمودار رسم شده است. همان طور که در شکل هم مشخص است مدل SkipTree بهبودیافته با تعداد گره‌های کم هم دارای هزینه‌ی تأخیر نسبی کمتری نسبت مدل پایه می‌باشد و در ضمن با افزایش تعداد گره‌ها هم هزینه تأخیر آن رشد کمتری نسبت به مدل پایه دارد در صورتی که مدل SkipTree پایه با هزینه‌ی تأخیر نسبی بیشتر در ابتدا شروع شده و شیب نمودار آن نیز بیشتر از مدل بهبودیافته است و با افزایش تعداد گره‌ها هزینه‌ی تأخیر آن هم رشد زیادی دارد.



شکل ۱: هزینه‌ی تأخیر نسبی برای هر دو نوع شبکه

۵ نتیجه

در این مقاله ما یک الگوریتم مسیریابی آگاه به هم‌بندی معرفی کردیم که این الگوریتم کارایی جدیدی را به داده‌ساختار SkipTree بهبودیافته اضافه می‌کند. این الگوریتم با در نظر گرفتن همسایگی در شبکه، عمل مسیریابی پیام‌ها را با توجه به هم‌بندی لایه فیزیکی شبکه انجام می‌دهد. ما در این مقاله با اضافه کردن جدول همسایگی و ارائه یک الگوریتم مسیریابی آگاه به هم‌بندی علاوه بر این که تمام ویژگی‌های داده‌ساختار قبلی را حفظ می‌کنیم، مسیریابی را هم با توجه به همسایگیِ گره‌ها در شبکه انجام می‌دهیم. نتایج ارزیابی و پیاده‌سازیِ الگوریتم جدید هم بهبود کارایی را در این مدل تأیید می‌کند. این داده‌ساختار جدید که داده‌ساختار «SkipTree بهبود یافته‌ی آگاه به هم‌بندی» است، علاوه بر کاهش هزینه تاخیر نسبی در شبکه باعث ایجاد محلیت مسیر، محلیت محتوا و مزایای امنیتی هم می‌شود. همان طور هم که در بخش نتایج تجربی نشان دادیم هزینه تأخیر نسبی این مدل جدید به ازاء هر گره نسبت به مدل قبلی کمتر است و همچنین شیب نمودار آن نیز نسبت به مدل قبلی از رشد کمتری برخوردار است.

مراجع

[۱] شاهپریان، جواد، **تجربیاتی در پیاده‌سازی و ارزیابی کارایی شبکه‌های P2P**، پایان نامه کارشناسی ارشد، دانشگاه صنعتی شریف، تهران، خیابان آزادی، ۱۰۱ صفحه، دی‌ماه ۱۳۸۵.

محمد، "**بهبود مسیریابی در داده‌ساختارهای توزیع‌شده‌ی مبتنی بر لیست‌های پرشی**"، دوازدهمین کنفرانس بین‌المللی انجمن کامپیوتر ایران، دانشگاه شهید بهشتی، اسفند ۱۳۸۵.

[3] I. Mirrezaei, Javad Shahparian, M. Ghodsi, *RAQNet: A Topology-Aware Overlay Network,* to appear, AIMS'2007, LNCS 4543 by Springer-Verlog, pp. 13-24, 2007.

[4] T. Papadakis. *Skip Lists and Probabilistic Analysis of Algorithms.* PhD thesis, University ofWaterloo, 1993. Also available as Technical Report CS93-28.

[5] S. Alaei, M. Toossi, and M. Ghodsi, *SkipTree: A Scalable Range-Queryable Distributed Data Structure for Multidimensional Data,* to appear, The 16th Annual International Symposium on Algorithms and Computation (ISAAC'2005), Dec. 19 - 21, 2005, Sanya, Hainan, China, LNCS 3827, Springer-Verlog, pp. 298-307, 2005.

[6] W. Pugh. *A Skip List Cookbook.* Technical Report CSTR-2286.1, University of Maryland, 1990.

[7] W. Pugh. *Skip Lists: A Probabilistic Alternative to Balanced Trees.* In Workshop on Algorithms and Data Structures, pages 437-449, 1989.

[8] H. Nazerzadeh, M. Ghodsi. *RAQ: A Range-Queriable Distributed Data Structure.* In Proc. the 31st Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2005), LNCS 3381, pp. 264-272, 2005.

[9] F. Kaashoek and D. R. Karger. *Koorde: A simple degree-optimal hash table.* In Proc. 2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS 2003), 2003.

tacharjee. *How to Model an Internetwork.* In Proceedings of IEEE Infocom'96, April 1996.

[15] J. Aspnes and G. Shah. *Skip Graphs.* In Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms, Jan. 2003.

[16] M. Naor, U.Wieder, *Know thy Neighbor's Neighbor: Better Routing for Skip-Graphs and Small Worlds,* the 3rd international workshop on peer to peer systems, San Diego, USA, February 26-27, 2004.

[17] (2001) Kazaa media desktop. Available: http://www.kazaa.com/

[18] (2001) Gnutella development forum, the gnutella v0.6 protocol. [Online]. Available: http://groups.yahoo.com/group/the gdf/files/

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. *A Scalable Content-Addressable Network.* In Proceedings of ACM SIGCOMM, Aug. 2001.

try: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), pages 329-350. Springer-Verlag, 2001.

[11] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. *Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications.* In Proceedings of the ACM SIGCOMM '01 Conference, pages 149-160, San Diego, California, August 2001.

[12] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. *Tapestry: A global-scale overlay for rapid service deployment.* IEEE Journal on Selected Areas in Communications, 2003. Special Issue on Service Overlay Networks, to appear.

[13] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron. *Topology-aware routing in structured peer-to-peer overlay networks.* Technical Report MSR-TR-2002-82, Microsoft Research, 2002.