



ارائه شده توسط:

سایت ترجمه فا

مرجع جدیدترین مقالات ترجمه شده

از نشریات معتبر

اولین تجربه با CarTorrent در بستر آزمون شبکه ادهاک خودروی حقیقی

چکیده:

محبوبیت تحقیقاتی که شامل اشتراک گذاری محتوایی با استفاده از مدل جفت با جفت (نظیر به نظیر) اشتراکی هستند در شبکه ad-hoc خودرویی بیشتر شده است. پنجره کوچک مربوط به حمل و نقل از یک وسیله نقلیه به یک نقطه دستیابی مورد نظر (AP) ، قابلیت جا به جایی یک وسیله و ارتباطات متناوب و دارای طول عمر کم به یک نقطه دستیابی مورد نظر دلایلی را برای داشتن مشارکت یک وسیله نقلیه با دیگری نشان میدهند که می توانند اطلاعاتی را از اینترنت کسب کنند. این گونه ویژگی های شبکه ad-hoc استفاده از نمونه های جفت اشتراکی را نشان میدهند و باعث ایجاد انگیزش برای ساخت برنامه های کاربردی مثل CarTorrent که محتوی را به اشتراک میگذارند می شوند.

ساختن چنین برنامه ای بر طبق تحقیقاتی در SPAPWN , CarTorrent را انجام دادیم و آن را بر روی VANET واقعی پیاده سازی کردیم. آزمایش های گسترده ای را در این زمینه اجرا کردیم تا میزان قابل اجرا بودن برنامه کاربردی که در خور VANET هستند ثابت شود. گسترش چنین برنامه های کاربردی وابسته به حمل و نقل ad-hoc اولین نوع از این مطالعات است.

1. معرفی

ابزارهای امنیتی راهبردی توسعه و ارتقا VANET را به سمت جلو برده است. در ورای امنیت راهبردی انواع جدیدی از برنامه های کاربردی مثل " آفیس آن ویلز " و " آن کار اینرنیتمن " پدید آمدند. در این میان ، اشتراک گذاری فایل دارد شتاب حرکتی خود را می یابد: مردم نه تنها در هنگام رانندگی میخواند موزیک و

ترلیر های فیلم ها را دانلود کنند بلکه میخواند داده هایی که مبتنی بر دانستن مکان فرد ستند مثل تور های مجازی هتل ها را نیز دانلود کنند.

مردم میتوانند این فایل ها را از طریق نقاط دستیابی کنار خیابانی دریافت کنند که اتصالات اینترنتی را فراهم می کند که با نام "وار درایوینگ" شناخته شده است. مدل کلاینت سرور مرسوم به دلایل زیر در این نوع اتصالات نه کار میکند و نه می تواند اندازه گیری کند. دلیل اول اینکه به خاطر قابلیت جابه جایی , زمان برقراری ارتباط واقعی با یک نقطه دستیابی کوتاه است. برای مثال با فرض اینکه رنج برد وای فای 300 متر است , زمانی که سرعت راننده 45mph است , ما می توانیم زمان اتصال 30 ثانیه ای داشته باشیم(در سی ثانیه ارتباط برقراری کنیم). با قرار دادن سر بار حاصل از اشتراک گذاری , DHCP , زمان واقعی کمتر از سی ثانیه خواهد بود. دلیل دوم اینکه . در محیط های واقعی طول سیگنال ها تابعی از فاصله می باشد . هنگامی که فاصله خودرو از نقطه دستیابی زیادتر می شود طول سیگنال کاهش می یابد. این امر باعث افزایش نرخ خطا به ازای هر پاکت می شود که به این ترتیب خروجی موثر که می تواند حاصل شود از چیزی که حاصل میشود کمتر است. دلیل سوم این است که در واقعیت نه این امر قابل اجرا اجرا است که در هر 300 متر یک نقطه دستیابی نصب کنیم نه آنکه بخواهیم در وسط راه ننگه داریم و دانلود کنیم ممکن است. پس این چنین در نظر میگیریم که در واقعیت زمان داشتن اتصال کم است و خروجی ایده آل کم. برای اینکه این موقعیت را به خوبی کنترل کنیم ما از هجومی که به اتصالات جفت با جفت شده است حمایت می کنیم که در آن کاربرانی که خارج از نقاط دستیابی هستند می توانند از یکدیگر قسمت هایی از فایل را دانلود کنند.

در فایل p2p مثل BitTorrent یک فایل به قطعاتی با اندازه مشابه تقسیم میشود و با قطعه های دیگر جفت می شود و می تواند با هر قطعه ی موجود دیگری با تشکیل دادن یک شبکه ی اضافی تعویض شود. این کار نه تنها بار سرور را کم میکند بلکه دسترسی به تکه ها را افزایش میدهد پس فرآیند دانلود کردن را نیز افزایش میدهد. هر چند BitTorrent بخاطر تفاوت بین توپولوژی مازاد و توپولوژی فیزیکی گره های قابل جا به جایی

نمیتواند مستقیماً به مناطقی که وایرلس در آنجا موجود است پورت شود. برای مثال جفتی که در توپولوژی منطقی به اندازه ی یک hop دور از ماست در توپولوژی فیزیکی پنج hop دور از ماست. برای ارتقا دادن تعداد منابع وایرلسی موجود توسط ترافیک مکانیابی (که باعث افزایش تنوع فاصله ای و کاهش سربار های مسیر یابی میشود) محققان بر روی نگاشت پوشش منطقی به توپولوژی فیزیکی پرداخت اند. به طور ویژه spawn از استراتژی "انتخاب بر مبنی تکه ی proximity driven" استفاده میکنند که میانگین hop را در کشیدن های چند hop ای کم کرده است. اینگونه تصور میشود که این استراتژی نسبت به روش مرسوم "کم یابترین اولین" کارایی بیشتری دارد .

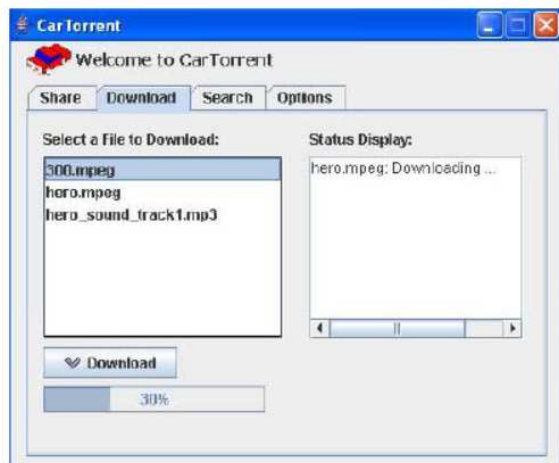
در این مقاله ما CarTorrent را ارائه می دهیم که پروتکلی است از سبک BitTorrent که در محیط های وابسته به حمل و نقل مورد استفاده است که توسط spawn توسعه داده شد. برای یک فایل کلاینت های CarTorrent تکه های اطلاعاتی موجودشان را با استفاده از روش پراکندن دیتا منتشر میکنند. (مثلاً k آمین hop با انتشار هدف محدود و مشخص میشود). هر انتشار پیام به سمت جلو هدایت می شود تا زمانی که به گره ای که در hop کی ام نسبت به مرکز نشر است برسد. سپس جفت ها می توانند آمار هایی را مثل توپولوژی محلی و تکه های موجود را جمع آوری کنند . این آمار ها برای انتخاب یک تکه/جفت که از نظر فاصله ای نزدیک تر هستند استفاده می شوند. به بیان دیگر اگر جفت آ و ب تکه کمیابی را که سی می خواهد در اختیار داشته باشند آنگاه سی می تواند توسط آ انتخاب شود چون آ با طی کردن hop کمتری می تواند به سی برسد. کاربران CarTorrent می توانند صف هایی را به کلاینت ها ارسال کنند. صف دارای تحمل در تاخیر است مثلاً زمانی که اتصال موجود است ارسال می تواند صورت بگیرد و مشکل حل شود. در نظر داشته باشید که خواستن اطلاعات دانلودی , جاهای دیدنی , پیش نمایش فیلم سینمایی مهم تر از میل به نگه داشتن ان اطلاعات می باشد و بنابراین انگیزه ی اشتراک گذاری را بیشتر فراهم می آورد.

هدف این مقاله امتحان کردن میزان عملی بودن اشتراک گذاری درون وسیله ای (وسیله نقلیه) به وسیله ی CarTorrent میباشد. برای پیش بردن این هدف ما برنامه CarTorrent را پیاده سازی میکنیم و کارایی آن را در بستر آزمایش VANET اندازه میگیریم. استفاده ی برنامه ی جفت به جفت در این آزمایشات اولین گام حرکت است. نشان میدهند و یم که جفت های اشتراکی میتوانند مکانسیم پراکندن را به بهره وری خوبی برسانند تا حضور یکی از جفت ها قابل تشخیص باشد استراتژی انتخاب قطعه برای دانلود کردن از دیگری به کار گرفته شود. ما آزمایشات را در سطح وسیعی انجام می دهیم و مقیاس های اندازه گیری کارایی را در VANET به دست می اوریم. مقایسات کارایی بین خط شروع پارکینگ های ساکن و سناریو های سیار جاده ای را نشان می دهیم. ما بر این باویم که بسیاری از درس هایی که اموخته ایم و تکنولوژی های که با این اندازه کاری برای آزمایش انتخاب شده اند برای این تحقیق ما بی اعتبار هستند.

ما بقی مقاله به صورت زیر برنامه ریزی شده است . قسمت دوم آرشیو CarTorrent و جزئیات پیاده سازی را نشان میدهد. قسمت سوم تجربه ی نصب ما و نتایج آن را ارائه میدهد . قسمت 4 کار های وابسته به آن را نشان میدهد. و نهایتا قسمت 5 نتیجه گیری های مقاله و کارایی احتمالی زمان فعلی و آتی را در این رابطه نشان میدهد.

۲.پیش نمایش و معماری CarTorrent

توالی اشتراک گذاری های جفت به جفت شراکتی به مفهومی به نام CarTorrent رسیده است که مانند پروتکل swarming با استفاده از مکانسیم پراکندگی و استراتژی جدید انتخاب قطعه از وایرلس متوسط و گره های مجاورتی بهره میبرد . در این مقاله به بررسی CarTorrent و معماری آن میپردازیم.



(a) File Downloading



(b) File Searching

شکل ۱. نمای گرافیکی برنامه CarTorrent.

2.1 بررسی اجمالی CarTorrent

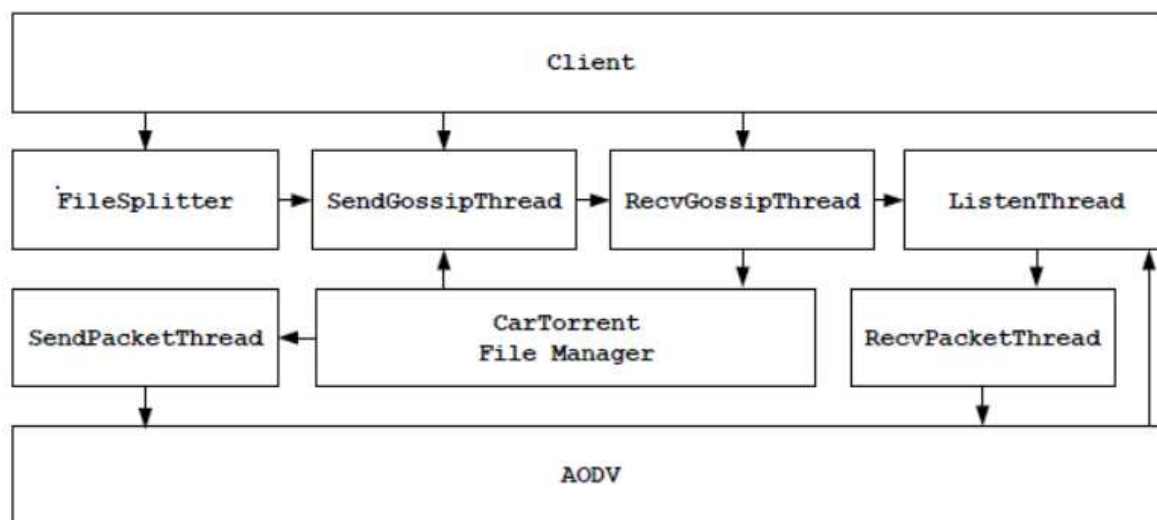
شکل a, b یک نمای GPU از CarTorrent را نشان می دهند. یک نوع معمولی از مورد استفاده آن به شرح زیر است: کلاینت آ می خواهد فایل را با F به اشتراک بگذارد. کلاینت آ توسط FileSplitter.SendGossipThread به چند تکه تقسیم کرده است و به صورت دوره ای gossip هایی را که شامل تولیدکننده فایل، شماره توالی، نام فایل، یک بیت قطاع از فایل، TTL و شمار HOP می باشد را ارسال میکند. به محض ارسال F gossip SendGossipThread هایی را مرتبط با وجود F ارسال میکند.

فایل بی می بیند که به عنوان RcvSendGossipThread آن به اشتراک گذاشته شده است و gossip های مربوط به F را دریافت میکند. سپس gossip ها به careTorrentFileManager فرستاده میشوند که ماشین الگوریتم انتخاب قطعه به حساب می آید. بر طبق این الگوریتم درخواست SendPacketThread درخواست میکند تا یک قطعه خاصی از کلاینت آ به وسیله AODV ارسال شود.

ListenThread از کلاینت آ درخواست های ورودی را خدمت دهی میکند. به محض دریافت قطعه خاصی

اتصال سریعاً با ReceivePacketThread پردازش میشود.

این خط قطعه را از محل ذخیره ایستای خود دوباره به دست می آورد و با AODV ارسال میکند. تفاوت بین ReceivePacketThread و SendPacketThread در آن است که SendPacketThread فایل را ارسال میکند و ReceivePacketThread فایل را دریافت میکند.



شکل ۲. معماری CarTorrent .

2.2 پیاده سازی CarTorrent

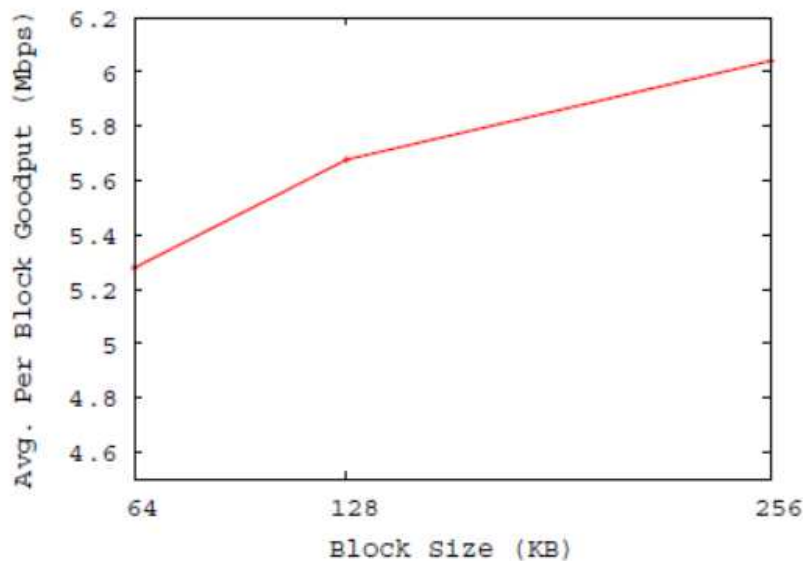
ما پنج مفهوم اصلی از CarTorrent را در شکل 2 در قسمت باقی مانده شرح میدهم.

2.2.1 فایل منجر CarTorrent

هر فایل در فایل منجر سازماندهی میشود. این بخش مسئول نگه داری رد حالت های هر قطعه از فایل است. بعلاوه لیست جفت ها را و عدد hop آنها را با هر قطعه نگه میدارد. اطلاعات حالات هر بار که قطعه ای دریافت میشود به روز رسانی میشود. لیست جفت ها و عدد هوب آنها هر باری که یک کلاینت gossip ای را دریافت میکند به روز رسانی میشود. بازیابی کمیابترین نزدیک ترین قطعه و جفتی که قطعه را دارا است به یک فایل

ListenThread و SendPacketThread 2.2.5

وقتی برنامه CarTorrent شروع میشود ListenThread برای اتصالات داخلی آن ساخته میشود. هر اتصال داخلی از ReceivePacketThread برای پروسه بعدی رد خواهد شد. به طور همزمان سیستم سه تا ReceivePacketThread میسازد که با مدل راند رابین اتصالات داخلی را پردازش میکنند. اگر تعداد نخ ها از سه تا تجاوز کند کجور از منتظر بماند و تاخیر بخورد. دو نوع پکت ورودی وجود دارد. اولین نوع درخواست دیتا میباشد. ReceivePacketThread با ارسال پکت درخواستی به این درخواست پاسخ میدهد. نوع دوم دیتا است. دیتای ورودی همانی قطعه فایلی است که گره درخواست کرده. این فایل در فضای محلی گره ذخیره شده و بعدا همه ی قطعات با هم ترکیب میشوند.



شکل ۴. خروجی پیش-قطعه در مقابل اندازه بلاک.

3. تجربه

در این قسمت دو سناریوی تجربی متفاوت را شرح می دهیم. سناریوی اول ارتباط استاتیک بین دو لپ تاپ در یک پارکینگ است. این سناریو به عنوان یک خط شروع مقایسه کارایی با دومین سناریو بکار میرود که در آن دو ماشین در حال حرکت قطعاتی را از هم دانلود که میکنند که در عین حال در یک مسیر مستقیم هستند.

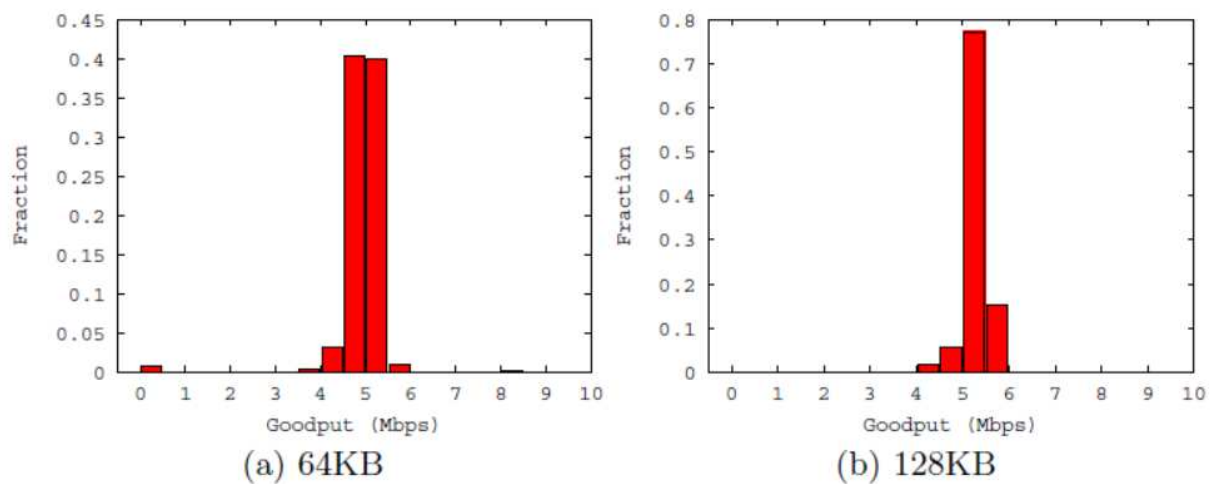
ما نتایج خط شروع و مسیر مستقیم را ارائه میدهیم. اول خط شروع برطبق قطعه در بهترین بازدهی را در مقابل سایز بلاک و میزان پراکندگی برطبق قطعه در بهترین بازدهی را بیان میکنیم. سپس پراکندگی قطعات دانلود شده در طی زمان مشخصی را با نشان دادن کیفیت ارتباط بین جفت ها با هم و بین جفت ها با جفت های دیگر در طی جاده مستقیم و نقطه دستیابی و برطبق قطعه در بهترین بازدهی مشخص را ارائه میدهیم .

3.1 سناریو ها

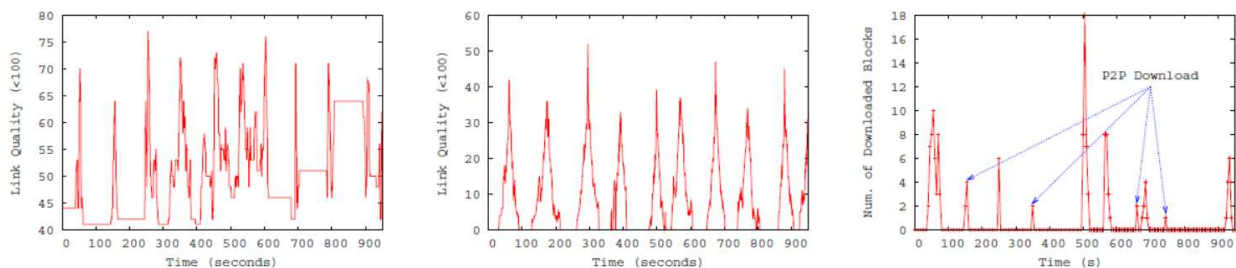
هر خودرو را به یک لپ تاپ با کارت interface وایرلس 802.11b مجهز کردیم. کارت interface یکی از آنها مسئول ارتباط بین خودرو ها با مود lad-hoc است. کارت دیگر برای ارتباط بین خودرو ها با نقطه دستیابی در مود پیدایش است. وقتی خودرویی به نقطه دستیابی دسترسی پیدا میکند gossip های و درخواست های قطعه ای را از interface دریافت میکند که gossip ها از آنجا آمده اند. بعد از مدتی برای تعامل با نقطه دستیابی این نقطه درخواست ها را دریافت میکند و قطعا درخواست شده را می فرستد. همزمان خودرو gossip هایی را دریافت میکند و قطعاتی را از جفت خود درخواست میکنند. از آنجایی که CarTorrent از نخ ها برای دریافت gossip های دریافتی استفاده میکند میتواند از شبیه ساز های gossip از هر دو interface مراقبت کند. درخواست ها به نام سازندگانشان مهر خورده اند. این امر به سیستم کمک میکند که راحت تشخیص دهد چه چیزی از چه کسی به چه کسی ارسال شود. به منظور جلوگیری از مداخلات کارت های interface ما بر روی کارت اولی کانال یک و بر روی دومی کانال 11 را نصب کرده ایم.

از پروتکل TCP استفاده کردیم. برای جلوگیری از هدر رفت در کال های مسدود کننده 0.65 SEC برای حداکثر زمان انتظار در نظر گرفتیم که ReceivePacketThread می تواند مدام به درخواست های ورودی پاسخ دهد. ما AODV و پیوند آن با CarTorrent را مورد آزمایش قرار دادیم.

دو نوع سناریو را در تجربیاتمان اجرا کردیم. اولین سناریو در یک پارکینگ زیر زمینی انجام شد تا مداخله های بین سیگنال ها برطرف شود.



شکل ۵: توزیع خروجی پیش قطعه (سناریوی پارکینگ).



شکل ۸. توزیع قطعات دانلود شده در طول زمان. شکل ۷. کیفیت پیوند بین نظیر ها در نقطه دستیابی مود

زیرساختی شکل ۶. کیفیت پیوند بین نظیر ها در اد هاک.

یک فایل 25MB از یک لپ تاپ به دیگری ارسال کردیم که به قطعات 64KB, 128KB, 256KB به ترتیب تقسیم شدند تا خط شروع دانلود و زمان کل خروجی بازای هر قطعه به دست آید.

سپس تجربه ی واقعی VANET مان را در یک جاده مستقیم 1KM کسب کردیم. (شکل 3)

نقطه دستیابی را در وسط جاده قرار دادیم.

هر دو خودرو از دو سر جاده شروع به حرکت کردند. دیدیم که ماشین ها از همدیگر و از نقطه دستیابی دانلود را انجام دادند. نتایجمان را در قسمت بعد نشان میدهیم.

3.2 نتایج تجربی

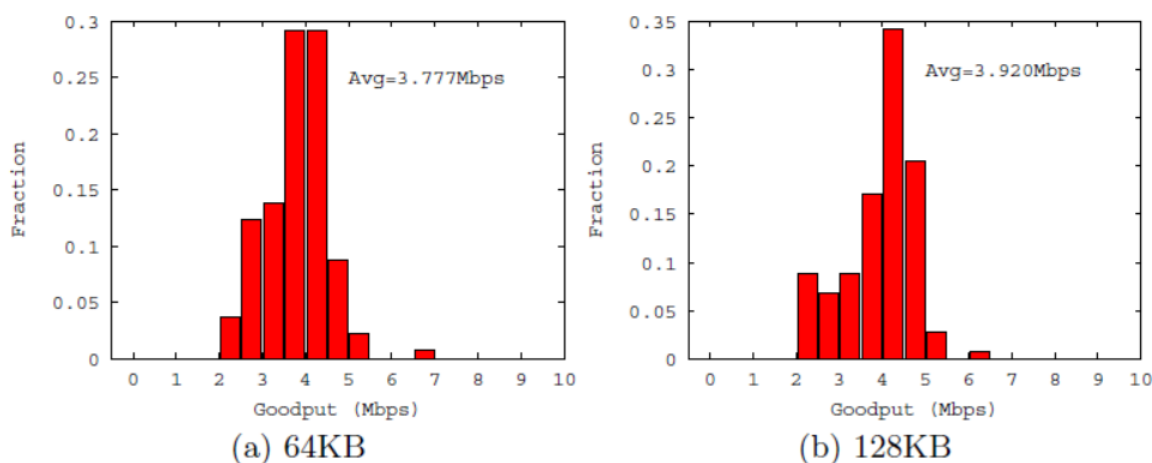
شکل 4 خروجی متوسط را با قطعاتی به حجم 64 , 128 , 256 KB نشان میدهیم. به دلیل آنکه قطعه ها از طریق TCP ارسال شده اند که زمانی را برای نصب اولیه میبرد بهتر آن است که با قطعات حجم های بالاتری تقسیم شوند. هر چند شیب چنین افزایشی در اندازه میانگین کاهش پیدا میکند زیرا قطعات بیشتری در یک شبکه وایرلس مستعد ارسال مجدد هستند. علاوه بر این بخاطر سایز بافر های ارسال کننده و دریافت کننده قطعات بزرگتری در معرض ریزش و پردازش شدن قرار داده میشوند. قطعاتی که ارسال نمیشوند بعد از قطع اتصال دور ریخته میشوند. همه ی این اوامر کم شدن میانگین خروجی بازای افزایش سایز قطعات را پیشنهاد میدهند.

3.2.2 سناریو ی جاده مستقیم

شکل 5 پراکندگی خروجی بازای 64 کیلو بایت و 128 کیلو بایت را نشان میدهند. برای قطعات 64 کیلو بایتی خروجی بازای قطعه 5.279 مگا بایت است. برای قطعات 128 کیلو بایتی این مقدار 5.677 مگا بایت. میزان تفاوت زمانی بین زمانی که درخواست ارسال میشود و میزان زمانی که دریافت میشود, خروجی بازای هر قطعه را تعریف میکند.

اشکال شش هفت و هشت ارتباط بین کیفیت پیوند و تعداد قطعات دانلود شده در یکی از دو خودرو ها را نشان میدهند. نتایج دانلود های جفت با جفت ها در چهار ردیف آورده شده اند که هر ردیف یک بازه زمانی را نشان میدهد. بخاطر کمبود فضای نگارش فقط نتایج یکی از ماشین ها را آوردیم. نتایج ماشین دیگر بسیار شبیه به همین ماشین می باشند. میله های کیفیت لینک با میله های تعداد دانلود شده ها بهم مربوط هستند. در باز 40-70 ثانیه و 520 تا 600 ثانیه هیچ فایلی دانلود نشده است. این همان دلیل طراحی استراتژی انتخاب قطعه را بازتاب میکند. دلیل خاصی برای چنین برنامه ای وجود ندارد و تصمیم گرفتیم تغییری در استراتژی بدهیم.

در بازه 780-810 و 890-910 هر چند کیفیت پیوند بین جفت ها خوب است ولی فعالیت دانلودی وجود ندارد. چنین استدلال کردیم که گره هیچ قطعه ای را نمیخواهد از جفتش درخواست کند. از آنجایی که هر جفت از قطعات یکسانی برخوردار هستند تغییر قطعاتی رخ نداد. این استدلال از آخرین نمودار 920-950 پشتیبانی میکند. در این حین همه ی قطعات از نقطه دستیابی دانلود شده اند. تاثیر زمان نصب TCP در زمان 480-520 مشاهده میشود. هر چند کیفیت پیوند نقطه و جفت بالا بود ولی تا زمان 510 دانلود کردن آغاز نشد. آخرین شروع اشتراک نقطه دستیابی و زمان نصب TCP را نشان میدهد.



شکل ۹. خروجی پیش قطعه در نظیر های توزیعی.

دو بازه زمانی خاص وجود دارد که در حدود 300 و 400 رخ میدهد. در حین این دو بازه کیفیت پیوند نقطه و جفت بالا بوده هر چند داندودی صورت نگرفته. ما نتایج را با گراف جفت دیگر مقایسه کردیم و این گونه برداشتیم که " تاثیر برداشت " از یکی از طرفین که درخواست میدهد باعث میشود تعداد نخ ها در نقطه دستیابی مشغول بررسی و خدمت رسانی به دیگر درخواست ها است. وقتی نقطه دستیابی آماده خدمت رسانی به جفت قحطی زده میشود آن جفت از رنج کاملا بیرون رفته است. همانطور که قبلا گفتیم تصمیم گرفتیم تا استراتژی انتخاب قطعه را تغییر دهیم تا جفت ها در رنج نقطه دستیابی داندود کنند. همینطور مایلیم تا افزایش تعداد نخ ها را در درخواست های ورودی تجربه کنیم تا قحطی زدگی کاهش یابد.

شکل 9 میانگین خروجی بازای قطعه را برای 64 کیلو بایت نشان میدهد. خروجی 1.5 مگا بایت کمتر از خط شروع پارکینگ است که در آن اختلال سیگنال وجود نداشت.

4 کار های مرتبط

BitTorrent به یک پروتکل مشهور برای ترافیک های اینترنتی تبدیل شده است. BitTorrent برای ad-hoc های شبکات سیار به خوبی توزین نمیشود چون اندازه های مشابهی را در پایه کار قرار میدهد پس تشخیص کمیاب ترین قطعاتی که دور یا نزدیک هستند را سخت میکند. این امر زمان داندود را درحالی که یک طرف مسیر طولانی تری را انتخاب کرده است افزایش میدهد تا به کمیاب ترین قطعه برسد. بخاطر سیار بود گره ها دور ترین گره ها کم معتبر ترین گره ها به شمار میروند. برای مشخصات یک VANET برای طراحی دیگری از BitTorrent فراخوانده میشوند. پروتکل هایی که شبیه BitTorrent هستند VANET جدید نیستند ولی قبلا فقط روی کاغذ شبیه سازی ها و تحلیل هایشان صورت گرفته. RandomWaypointModel سیار بودن و حرکت یک خودرو را در ns-2 شبیه سازی میکند. SPAWN, همان چیزی که این کار در سدد توسعه آن است فقط شبیه سازی را در Nab که یک شبکه شبیه ساز است انجام داده. ما torrent

swarming را بر طبق SPAWN در فضای آزمایشی VANET پیاده سازی کردیم. علاوه بر این پیاده سازی مسیر یابی AODV در لینوکس جا داده شد.

codTorrent در مرجع ۴ پروتکل نظیر به نظیر دیگری برای اشتراکگذاری فایل ها است که قطعات فایل را با کدگذاری عملیات ها مبادله میکند که تصادفی بود قطعات کد گذاری شده را بکار میگیرد و از این رو زمان دانلود در ونت را کاهش میدهد. برخلاف carTorrent یک نظیر فقط لازم است هرکدام از k قطعه کد گذاری شده ای را برای کدگشایی دریافت کند که در آن ، k تعداد قطعات یک فایل است. افزونگی ذاتی کد گذاری شبکه از مسائل حرکت گره ها مراقبت میکند (۲). بعلاوه ، گره ها از بلاک های کد شده سربار به نحوی سود میبرند هر چند مستقیما در گیر ارتباطات نمیشوند. این مسئله باعث افزایش مکان های k بلاک کد شده می شود و از این رو به زمان دانلود سرعت دهی میشود. استراتژی CodeTorrent برای انتشار قطعه ها با استراتژی CarTorrent متفاوت است. تصمیم گرفته شد تا CarTorrent در دنیای واقعی ونت پیاده سازی شود و نتایج آن با نتایج CarTorrent مقایسه شود.

به طور خلاصه، کار ما خاص است زیرا پیاده سازی واقعی CarTorrent و معیار های کاری آن در بستر آزمون انجام میشود.

۵. نتیجه گیری

شبکه های اد هاک خودرویی دارند در حال افزایش هستند و پیاده سازی آنها در مناطقی که شبکات سنسوری دارند و تبلیغات سیل آسا انجام میدهند نیز در حال افزایش است. در این مقاله، تمرکز ما بر روی متنوع تراشی مسئله ی پروتکل اشتراک گذاری محتوی نظیر به نظیر برای گره های خودرویی در ونت واقعی است. تشخیص فایل های موجود با gossip های دوره ای و فایل های دانلودی موفق نشان داده شد. انتخاب کارآمد یک قطعه فایب با استفاده از استراتژی Rarest-Closest First ممکن است که در آن ابتدا نایاب ترین قطعه فایب

مشخص میشود و لازم است تا نزدیک ترین گره به آن قطعه پیدا شود. این استراتژی از جابجای خودرویی گره ها در پیوند های بی سیم مستعد خطا بهره وری میکند.

در نهایت، CarTorrent را بر روی بستر آزمون ادهاک خودرویی پیاده سازی کردیم و بسط دادیم. حقایق امید بخشی را در مورد اجرای برنامه های اشتراک محتوی نظیر به نظیر نشان دادیم که مناسب محدودیت های ونت هستند. برنامه داریم تا تاثیرات شرایط شبکه را با مکان های مختلفی از نقاط دستیابی و با مکانیسم gossip را با یکی کردن جهت ها، سرعت و اطلاعات فاصله ای مطالعه کنیم .



این مقاله، از سری مقالات ترجمه شده رایگان سایت ترجمه فا میباشد که با فرمت PDF در اختیار شما عزیزان قرار گرفته است. در صورت تمایل میتوانید با کلیک بر روی دکمه های زیر از سایر مقالات نیز استفاده نمایید:

لیست مقالات ترجمه شده ✓

لیست مقالات ترجمه شده رایگان ✓

لیست جدیدترین مقالات انگلیسی ISI ✓

سایت ترجمه فا ؛ مرجع جدیدترین مقالات ترجمه شده از نشریات معتبر خارجی