



دانشگاه علم و صنعت ایران

مرکز آموزش های الکترونیکی

گروه مهندسی فناوری اطلاعات و ارتباطات

شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2

پروژه درس شبکه های کامپیوتری پیشرفته

اساتید درس: دکتر فتحی

مهندس غفاریان

تهیه کنندگان: داریوش اکبری تولون

آزیتا پلوان


علیرضا قاضی سعیدی

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

چکیده:


هدف از این گزارش ارائه نمونه های شبیه سازی شده شبکه های VANET با استفاده از شبیه ساز NS2 بوده است. شبکه های VANET نوعی از شبکه های Ad_hoc هستند که ارتباط بین وسایل نقلیه مجاور، همچنین بین وسایل نقلیه و تجهیزات ثابتی که معمولاً کنار جاده ها نصب می شوند را فراهم می آورند، هدف اصلی در شبکه های VANET ایجاد راحتی و امنیت بیشتر برای مسافران می باشد که از طریق هشدار تصادف، اعلان شرایط ترافیکی و علائم رانندگی و در جاده ها و سایر موارد مشابه انجام می شود. شبکه های VANET موضوعات تحقیقاتی جدید و رو به گسترشی را ایجاد کرده اند، از آنجایی که ایجاد محیط واقعی برای بررسی موضوعات مختلف مربوط به این نوع شبکه ها با مشکلات و پیچیدگی های بسیاری مواجه خواهد بود و نیز هزینه زیادی را می طلبد استفاده از شبیه سازهای مطرح از جمله NS2 روش مناسبی برای حل این نوع مشکلات می باشد.

در این گزارش ضمن ارائه توضیحات جامعی در خصوص شبیه ساز NS2، قابلیت ها، نحوه نصب و شبیه سازی با آن، شبکه های VANET مورد بررسی قرار گرفته و چند نمونه شبیه سازی شده برای این شبکه ها ارائه شده است.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---

فهرست مطالب

عنوان	شماره صفحه
فصل ۱- معرفی شبیه ساز NS2	۶
۱-۱- مقدمه	۷
۱-۲- نصب شبیه ساز	۷
1-2-1- محیط مورد نیاز و نحوه نصب N.S.	۷
1-2-2- دانلود کردن N.S.	۷
1-2-3- نصب N.S.	۷
۱-۲-۴- نصب NS2 در محیط ویندوز	۱۰
1-3- شروع کار با N.S.	۲۶
1-4- شبیه سازی N.S.	۲۷
1-4-1- OTCL زبان مورد استفاده کاربر	۲۹
1-4-2- یک مثال ساده از شبیه سازی	۳۱
1-4-3- زمان بند رخداد (Event Scheduler)	۳۷
1-4-4- اجزای شبکه Network components	۳۹
1-4-5- گره و مسیریابی	۴۰
1-4-6- ردیابی (Tracing)	۴۱
1-4-7- بسته	۴۳
1-4-8- مثال تجزیه و تحلیل ردیابی	۴۴
1-4-9- مثالی از نمایشگر صف RED	۴۷
1-4-10- چه چیزی را کجا پیدا کنیم؟	۵۰
1-4-11- پیوند OTCL	۵۲
1-4-12- صادر کردن کلاس از C++ به OTCL	۵۲
1-4-13- صادر کردن متغیر از C++ به OTCL	۵۳
1-4-14- اضافه کردن یک کاربرد و یک عامل (Agent) جدید	۵۷
1-4-15- اضافه کردن یک صف جدید	۶۲
1-4-16- مثال LAN	۶۵
1-4-17- مثال چندپخش (Multicasting)	۶۷
1-4-18- مثال وب سرور	۶۹
1-5- معرفی برنامه NAM	۷۲


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۶- دستورات اولیه در NS2	۷۶
فصل ۲- معرفی شبکه های VANET	۸۲
۱-۲- مقدمه	۸۳
۲-۲- بررسی شبکه های VANET	۸۳
3-2- تکنولوژی ها و پروتکل های شبکه های VANET	۸۵
فصل ۳- شبیه سازی شبکه های VANET در NS2	۸۷
۱-۳- مقدمه	۸۸
۲-۳- نمونه شبیه سازی شده ۱	۸۹
۱-۲-۳- طرز اضافه کردن یک پروتکل جدید به NS-2	۸۹
۲-۲-۳- برنامه tcl با چهار گره موبایل	۹۶
3-3- نمونه شبیه سازی شده ۲	۱۰۷
۴-۳- نمونه شبیه سازی شده ۳	۱۱۲
۱-۴-۳- پیکربندی	۱۱۹
۲-۴-۳- مراحل شبیه سازی	۱۲۰
۳-۴-۳- پیاده سازی یک مثال	۱۲۲
3-4-4- تولید نقشه	۱۲۶
۵-۴-۳- تولید حرکت خودروها	۱۳۴
۶-۴-۳- تعریف جریان	۱۳۴
۷-۴-۳- تعیین دور	۱۳۶
۸-۴-۳- حرکت خودکار خودروها	۱۳۸
۹-۴-۳- تنظیمات شبیه سازی	۱۳۹
منابع و مراجع	۱۴۵


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

فهرست اشکال


شکل ۱: اجزای کلی ns.....	۲۸
شکل ۳: توپولوژی یک شبکه ساده	۳۴
شکل ۴: زمان بند رخداد گستر	۳۷
شکل ۵: بخشی از سلسله مراتب کلاس	۳۹
شکل ۶: گره چندپخشی و تک پخشی	۴۰
شکل ۷: لینک ساده.....	۴۱
شکل ۸: وارد کردن شیئی های ردیابی	۴۲
شکل ۹: مشاهده صف	۴۲
شکل ۱۰: مثالی از حرکت بسته	۴۳
شکل ۱۱: فرمت بسته در NS	۴۴
شکل ۱۲: مثالی از فرمت ردیابی	۴۵
شکل ۱۳: لزش CBR در گره n3	۴۷
شکل ۱۴: تنظیمات مربوط به مثال نمایشگر صف RED	۴۸
شکل ۱۵: گراف ردیابی صف Red	۵۰
شکل ۱۶: ساختار شاخه ها در NS	۵۱
شکل ۱۷: مثال C++ از جزء شبکه و شیئی پیوند	۵۳
شکل ۱۸: مثالی از تخصیص متغیرها	۵۳
شکل ۱۹: مثالی از مفسر دستور OTCL	۵۴
شکل ۲۰: اجرای دستور OTCL از شیئی C++	۵۴
شکل ۲۱: ex.linkage.tcl	۵۶
شکل ۲۲: نتیجه مربوط به برنامه ex.linkage.tcl	۵۶
شکل ۲۳: ساختار سرآمد MM و کلاس آن	۵۸
شکل ۲۴: اضافه کردن به فایل packet که از نوع C++ است	۵۹
شکل ۲۵: اضافه کردن به فایل ns-packet.tcl که از نوع OTCL است	۵۹
شکل ۲۶: ارسال اجرای تایمر	۶۰
شکل ۲۷: افزودن دو تابع محلی به کلاس Agent	۶۱
شکل ۲۸: افزودن دو تابع محلی به کلاس Application	۶۱
شکل ۲۹: تنظیم مقادیر اولیه پارامترها	۶۱
شکل ۳۰: توپولوژی و سناریوی شبیه سازی برنامه Test	۶۲
شکل ۳۱: پیاده سازی کلاس DtRr Queue	۶۵
شکل ۳۲: برنامه آزمایش Dr Rr Queue	۶۵
شکل ۳۳: توپولوژی شبکه LAN و شبیه سازی آن	۶۷
شکل ۳۴: صفحه NAM مربوط به شبیه سازی ارسال چندپخشی	۶۹
شکل ۳۵: توپولوژی شبکه web server و شبیه سازی آن	۷۱

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۷۳.....	شکل ۳۶: معرفی محیط ابزار Network Animator.....
۷۵.....	شکل ۳۷: خروجی Tcl Code در Nam . بسته ها بصورت پیکانهایی نمایش داده میشوند.....
۸۱.....	شکل ۳۸: مونیتور کردن یک لینک.....
۱۰۰.....	شکل ۳۹: محیط نرم افزار xgraph.....
۱۰۱.....	شکل ۴۰: باز کردن فایل Trace.....
۱۱۳.....	شکل ۴۱: صفحه دانلود نرم افزار sumo.....
۱۱۴.....	شکل ۴۲: محیط نرم افزار sumo.....
۱۲۲.....	شکل ۴۳: ارتباط نرم افزار SUMO و NS2 و MOVE.....
۱۲۵.....	شکل ۴۴: انتخاب سیستم عامل سیستم.....
۱۲۶.....	شکل ۴۵: مشخص کردن مسیر نرم افزار sumo در محیط نرم افزار move.....
۱۲۶.....	شکل ۴۶: ایجاد نقشه به صورت دستی.....
۱۲۷.....	شکل ۴۷: وارد کردن گرورها به صورت دستی.....
۱۲۸.....	شکل ۴۸: انتخاب منوی Edge از منوی اصلی.....
۱۲۸.....	شکل ۴۹: پنجره ظاهر شده بعد از انتخاب edg.....
۱۳۳.....	شکل ۵۰: تولید نقشه.....
۱۳۴.....	شکل ۵۱: پنجره ظاهر شده برای تعریف جریان.....
۱۳۵.....	شکل ۵۲: پنجره ظاهر شده پس از کلیک کردن بر روی flow.....
۱۳۶.....	شکل ۵۳: تعیین دور در منو اصلی.....
۱۳۷.....	شکل ۵۴: پنجره ظاهر شده پس از کلیک بر روی Turn.....
۱۳۸.....	شکل ۵۵: حرکت خودکار خودروها.....
۱۳۸.....	شکل ۵۶: پنجره ظاهر شده پس از کلیک بر روی create vehicle.....
۱۳۹.....	شکل ۵۷: کلیک بر روی گزینه configuration.....
۱۳۹.....	شکل ۵۸: پنجره ظاهر شده پس از کلیک بر روی configuration.....
۱۴۰.....	شکل ۵۹: پنجره ظاهر شده پس از کلیک بر روی Visualization.....
۱۴۱.....	شکل ۶۰: انتخاب برای مدل ترافیک.....
۱۴۱.....	شکل ۶۱: مدل ترافیک نرم افزار ns2.....
۱۴۲.....	شکل ۶۲: خروجی برای ns2 تولید کردن.....
۱۴۳.....	شکل ۶۳: وارد کردن اطلاعات مربوط به شبیه ساز ns2.....
۱۴۳.....	شکل ۶۴: وارد کردن اطلاعات در NS2.....
۱۴۴.....	شکل ۶۵: خروجی در شبیه ساز NAM.....

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

معرفی شبیه ساز NS2

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

۱-۱- مقدمه

در این فصل از گزارش به معرفی شبیه ساز NS2، طریقه نصب آن در محیط لینوکس و ویندوز پرداخته می شود، همچنین در رابطه با قابلیت های شبیه ساز، دستورات ساده آن، برنامه نویسی TCL و سایر موارد توضیحات جامعی در قالب مثال های مختلف ارائه شده است.

۱-۲- نصب شبیه ساز

در این فصل کلیاتی در مورد چگونگی نصب این نرم افزار خواهیم داشت. در طراحی این شبیه ساز از زبان برنامه نویسی C++ استفاده شده است و همچنین از زبان otcl نیز به عنوان واسط و مترجم فرامین استفاده می شود. به علت سرعت بالای C++، از آن برای پیاده سازی پروتکل ها و پردازش بسته های اطلاعات ورودی استفاده می شود. اما برای شبیه سازی ساختار و توپولوژی شبکه از زبان otcl استفاده می شود.

۱-۲-۱- محیط مورد نیاز و نحوه نصب N.S.


شبیه ساز N.S. بر روی سیستم های عامل مختلف UNIX نظیر Sunos و Linux قابل نصب می باشد. البته می توان از سیستم عامل windows نیز استفاده نمود، ولی N.S. ذاتاً برای محیط های UNIX طراحی شده است.

۱-۲-۲- دانلود کردن N.S.

به دو طریق می توان از برنامه N.S. استفاده کرد. روش اول استفاده از بسته های نرم افزار مختلف (مانند TCL/TK و otcl و ...) و روش دوم، استفاده از بسته نرم افزاری all-in-one می باشد. پیشنهاد می کنیم برای شروع کار با این برنامه از بسته نرم افزاری all-in-one استفاده کنید. تنها عیب آن این است که ممکن است بعضی از اجزاء بسته all-in-one مورد استفاده شما نباشد و فضایی از دیسک را اشغال کنند. البته توجه به این نکته ضروری است که بسته نرم افزاری all-in-one فقط در سیستم های بر مبنای Unix کار می کنند. این نرم افزار را می توان از آدرس اینترنتی <http://www.isi.edu.nsnam/dist/> دانلود کرد.

۱-۲-۳- نصب N.S.

به روش های مختلفی می توانید بسته نرم افزاری all-in-one را در اختیار داشته باشید. اگر بسته نرم افزاری N.S. را با استفاده از سیستم عامل windows، download کرده اید به طوریکه سورس این نرم افزار روی پارتیشن های windows می باشد، می توانید با استفاده از دستور mount به پارتیشن های ویندوز دسترسی پیدا کرده و مراحل نصب را انجام دهید. اگر بخواهید سورس بسته نرم افزاری N.S. را روی پارتیشن های پیدا

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

کرده و مراحل نصب را انجام دهید. اگر بخواهید سورس بسته نرم افزاری N.S. را روی پارتیشن های Linux داشته باشید، باید تنظیمات Internet Explorer محیط Linux را انجام داده و آن را مستقیماً در سیستم عامل Linux، دانلود کنید. همچنین روش سومی نیز وجود دارد که سورس این نرم افزار را روی cd داشته باشید که در این حالت باید CDROM را mount کرده و از سورس نرم افزار برای نصب آن استفاده کنید (در واقع در این حالت باید سورس مربوط را روی پارتیشن های Linux کپی کنید). حال با توجه به اینکه یکی از سه روش بالا به این نرم افزار دسترسی پیدا کرده اید، مراحل نصب به صورت زیر خواهد بود:

ابتدا فایل ns-allinone-2.1b6.tar (یا نسخه های بالاتر آن) که به صورت فشرده شده است را باید از حالت فشرده خارج کنید. اصطلاحاً باید آن را untar کنید، که در سیستم عامل Linux این کار با دستور زیر انجام می شود:

```
Tar-xzvfns-allinone-2.1 b6.tar
```

فایل های مزبور از حالت فشرده خارج می شوند. دستورات زیر را اجرا کنید تا برنامه نصب شروع به کار کرده و این نرم افزار را روی سیستم نصب کند.

```
Cd ns-allinone-2.1b6
./install
```

بعد از اتمام نصب، در فایل bash-profile، طبق دستوراتی که در انتهای نصب بر روی صفحه مانیتور ظاهر می شود، کلید مسیری را تنظیم کنید. فایل های اجرای برنامه N.S. و Nam در شاخه bin وجود دارند. در نهایت باید عبارت مقابل را به دستور path در فایل فوق اضافه کنید تا مراحل نصب پایان یابد.

```
Root/ns-allinone-2.1 b6/bin
```

توجه به این نکته ضروری است که جهت آنکه شبیه ساز N.S. به طور صحیح اجرا گردد، باید از نسخه 7.5 یا بالاتر استفاده شود. همچنین چنانچه از gcc به عنوان کامپایلر C++ استفاده می شود، باید از نسخه 2.7.2 یا بالاتر استفاده نمود. بعد از نصب N.S. برای اطمینان از صحت عملیات نصب و تست سیستم، می توان در بالاترین شاخه، دستور validate را اجرا نمود. این دستور یک سری تست های استاندارد از قبل تهیه شده برای N.S. را اجرا می کند و گزارشی از خروجی هر تست بر روی صفحه مانیتور اعلام می شود. برنامه تست validate پروتکل های زیر را تست می نماید:

- در سطح لایه کاربردی
 - پروتکل HTTP جهت اتصال به web.
 - پروتکل های telnet و FTP.
 - منابع ترافیکی از نوع نرخ بیت ثابت (CBR).
 - منابع ترافیکی از نوع ON/OFF.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

• در سطح لایه ارسال

- پروتکل TCP پایه.
- انواع پروتکل های TCP شامل: Full، FACK، SACK، Neew-Reno، Reno و Tahoe
- پروتکل های SRM و RTP.

• پروتکل های مسیریابی


- مسیریابی الگوریتمی
- مسیریابی سلسله مراتبی
- مسیریابی شبکه های محلی و پراکنشی (Broadcast)
- مسیریابی Manual
- مسیریابی Multi cast
- مسیریابی دینامیکی
- شبیه سازی در سطوح لایه جلسه (Session Level Simulation)

• مکانیسم های مسیریابی

- چندین الگوریتم زمان بندی صف شامل الگوریتم های:
 - FQ (Fair Queuing)
 - DRR (Deficit Round Robin)
 - SFQ (Stochastic Fair Queuing)
 - FIFO (First In First out)
 - CBQ (Class Based Queue)
- مدیریت صف از نوع RED
- الگوریتم های کنترل دسترسی شامل: MS, HB, ACTP, ACTD

• مکانیزم های لایه پیوند داده


- شبکه های محلی LAN با پروتکل CSMA/CD
- در شبیه ساز N.S. علاوه بر پروتکل های فوق که ذکر گردید، پروتکل های زیر نیز موجوداند که در برنامه تست Validate در نظر گرفته نشده اند:
 - پروتکل ASYMTCP
 - پروتکل های RLM, SNOOP, RTP, RTCP
 - شبکه های محلی با پروتکل های CSMA/CA
 - فیلترهای Token Bucket
 - منابع Trace-generated
 - گیرنده های Delay-adaptive
 - ماژول های تأخیر
 - IVS

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۲-۴- نصب NS2 در محیط ویندوز

برای نصب ns در ویندوز نیاز است ابتدا برنامه ای به نام cygwin نصب گردد، تا محیط لینوکس را در ویندوز شبیه سازی کنیم. بدین منظور ابتدا برنامه cygwin را نصب می کنیم. صفحه اینترنتی <http://www.cygwin.com/> را باز می کنیم.

[Cygwin Home](#)
[Cygwin/X Home](#)
[Red Hat Cygwin Product](#)
[Community](#)
[Reporting Problems](#)
[Mailing Lists](#)
[Newsgroups](#)
[Gold Stars](#)
[Mirror Sites](#)
[Donations](#)
[Documentation](#)
[FAQ](#)
[User's Guide](#)
[API Reference](#)
[Acronyms](#)
[Contributing](#)
[Snapshots](#)
[Source in CVS](#)
[Cygwin Packages](#)
[Software](#)
[Setup Package Search](#)
[Related Sites](#)
[Licensing Terms](#)
[sourceware.org](#)



What Is Cygwin?

Cygwin is a Linux-like environment for Windows. It consists of two parts:

- A DLL (cygwin1.dll) which acts as a Linux API emulation layer providing substantial Linux API functionality.
- A collection of tools which provide Linux look and feel.


The Cygwin DLL currently works with all recent, commercially released x86 32 bit and 64 bit versions of Windows, with the exception of Windows CE.

Note that the official support for Windows 95, Windows 98, and Windows Me will be discontinued with the [next major version \(1.7.0\)](#) of Cygwin, which is in [beta testing](#) right now.

What Isn't Cygwin?

- Cygwin is **not** a way to run native linux apps on Windows. You have to rebuild your application *from source* if you want it to run on Windows.
- Cygwin is **not** a way to magically make native Windows apps aware of UNIX ® functionality, like signals, ptys, etc. Again, you need to build your apps *from source* if you want to take advantage of Cygwin functionality.

[Help, contact, web page, other info...](#)


[Install or update now!](#)
(using setup.exe)

or

[get help](#) on using setup.exe.

or

[find](#) where a package or file lives in the Cygwin release.

Latest Cygwin DLL release version is [1.5.25-15](#)


Installing and Updating Cygwin


The latest net releases of the Cygwin DLL are numbered *1.n.x*, where "n" is currently "5" (e.g., 1.5.25). Any Cygwin program built from December 1998 onward should work correctly with 1.n.x DLLs.


The *1.n.x* version numbering refers only to the Cygwin DLL. Individual packages like *bash*, *gcc*, *less*, etc. are released independently of the DLL. The [setup.exe](#) utility tracks the versions of all installed components and provides the mechanism for **installing** or **updating** everything available from this site for Cygwin.

The [signature](#) for [setup.exe](#) can be used to verify the validity of this binary using [this](#) public key.


Run [setup.exe](#) any time you want to update or install a Cygwin package.

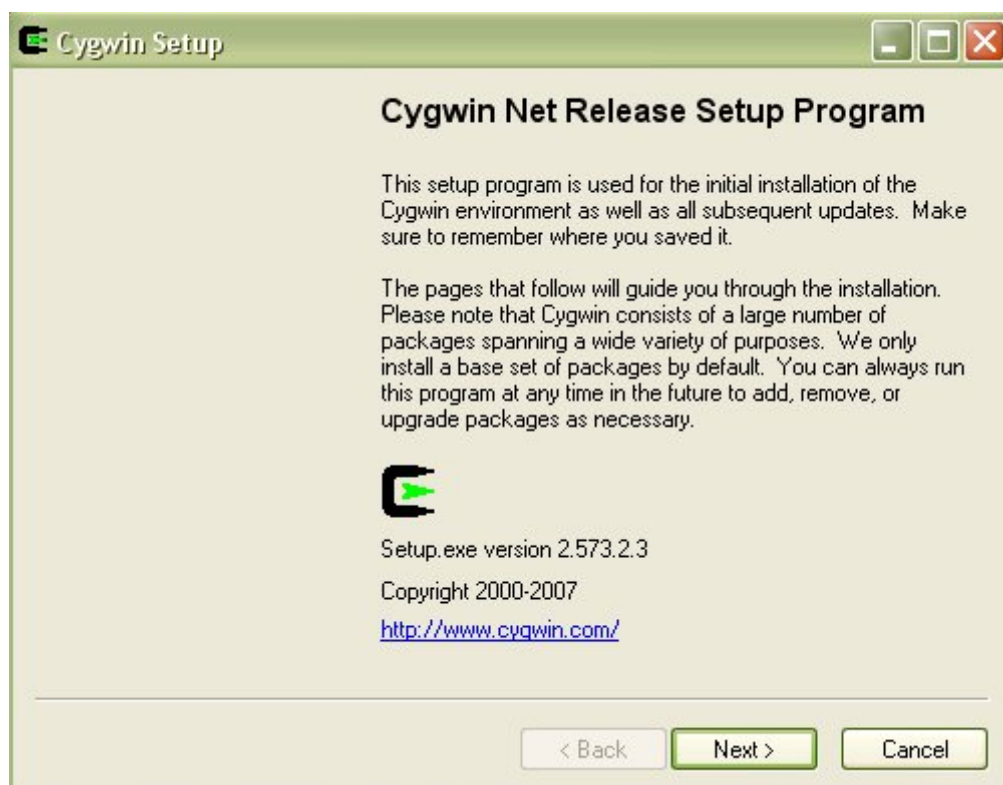




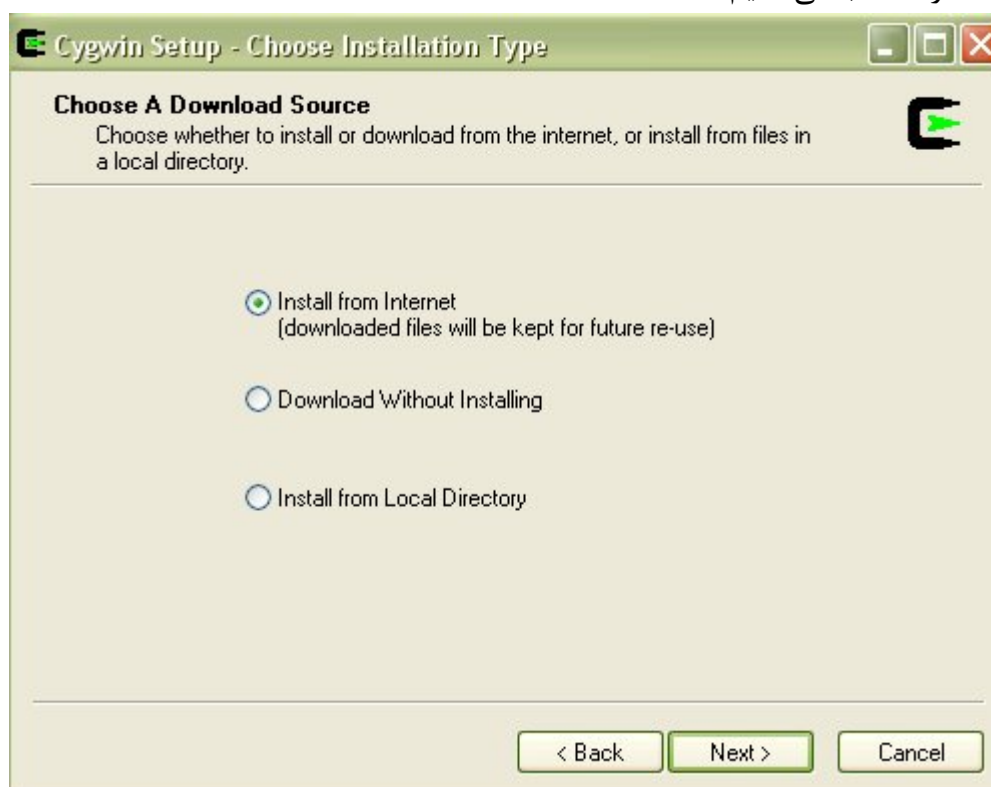


روی لینک install cygwin now که در گوشه سمت بالای سمت راست صفحه قرار دارد کلیک نموده، در این مرحله پنجره زیر ظاهر می شود.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

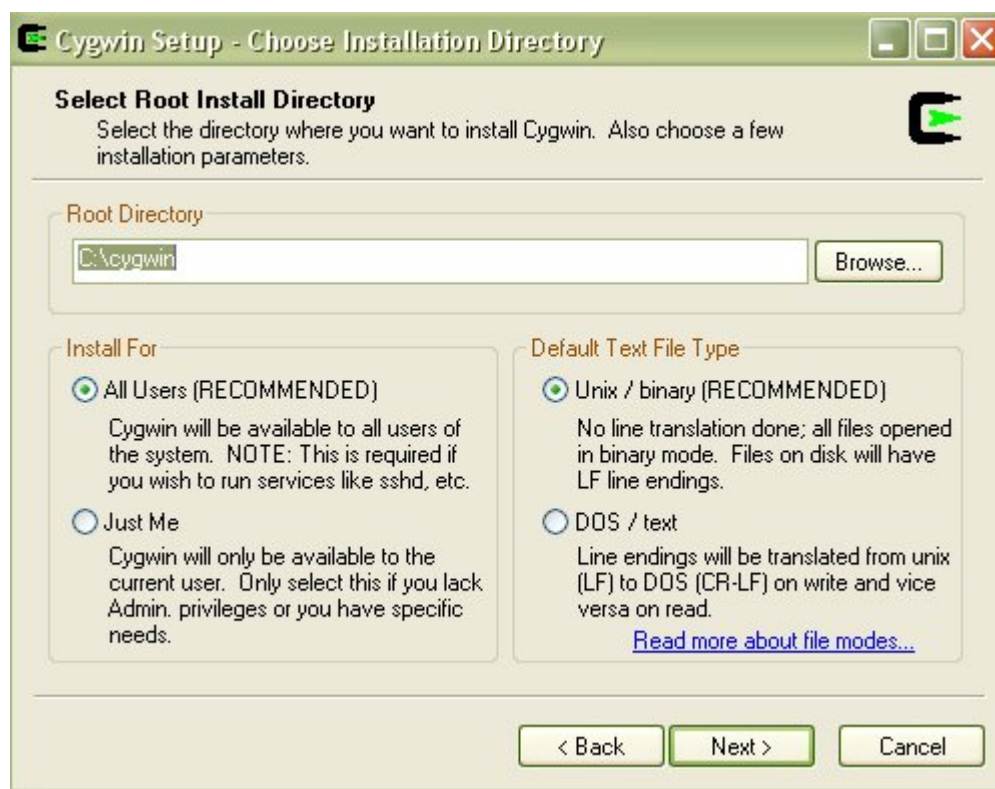


گزینه next را انتخاب می کنیم.




	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

گزینه install from internet را انتخاب می کنیم. این گزینه فایل های مورد نیاز برای نصب cygwin را در پوشه مورد نظر دانلود خواهد کرد. حال دکمه next را فشار می دهیم. در این صورت پنجره زیر خواهد آمد.




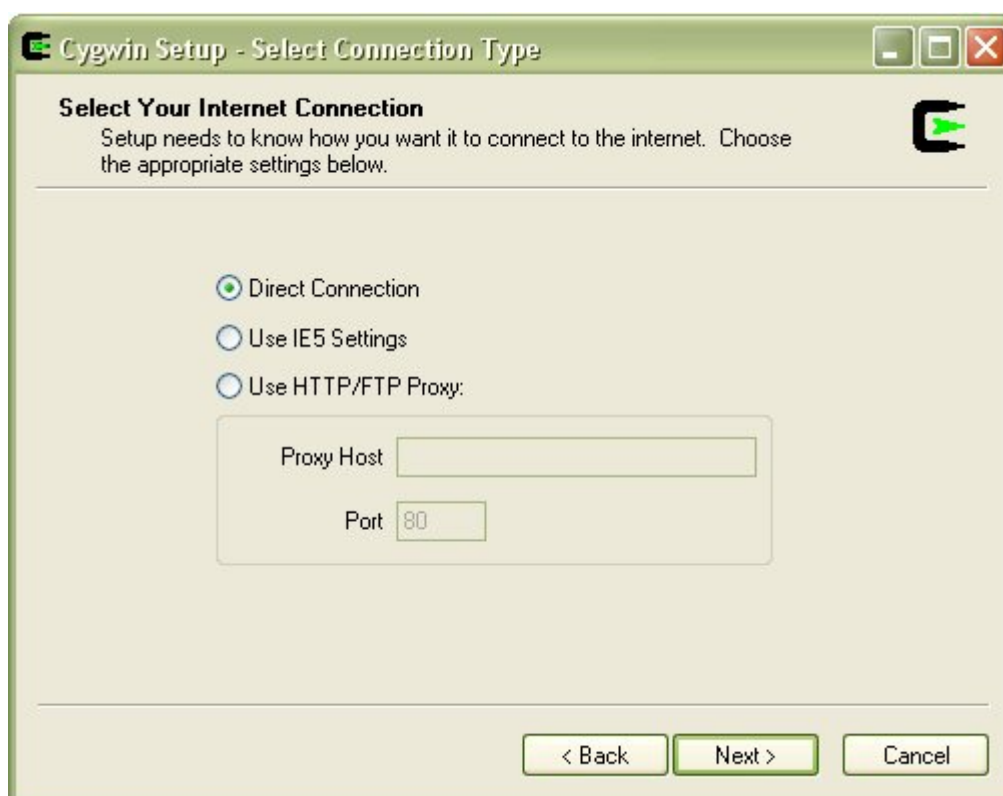
پوشه ای را که قرار است cygwin در آن نصب شود در قسمت root directory می نویسیم. پیشنهاد می شود که این گزینه را نیز به صورت پیش فرض خود رها کنیم. سایر گزینه ها را نیز بصورت پیش فرض رها کرده و دکمه next را انتخاب می کنیم. سپس پنجره زیر را مشاهده می کنیم.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---




در این مرحله باید نام و مکان پوشه ای را انتخاب کنیم که فایل cygwin.exe در آنجا ذخیره خواهد شد. سپس دکمه next را انتخاب می کنیم. در این حالت پنجره زیر ظاهر می شود. اگر برای ارتباط با اینترنت از proxy استفاده می کنیم باید در این پنجره تنظیمات لازم را انجام دهیم در غیر این صورت گزینه direct connection را انتخاب می کنیم.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---




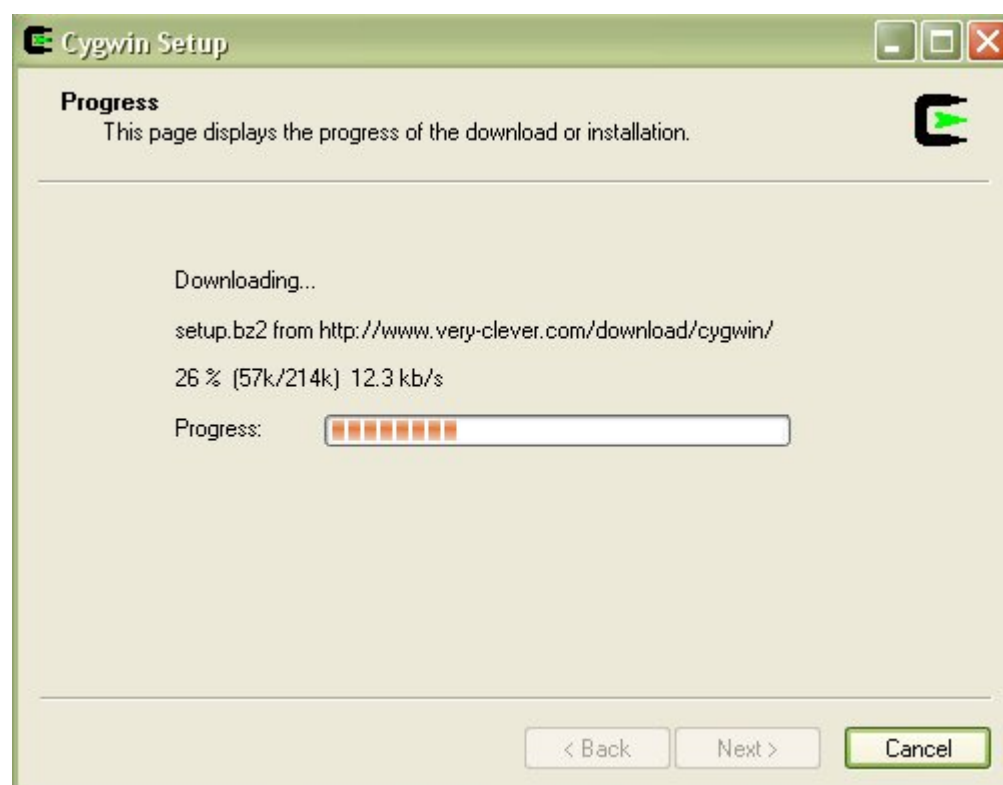
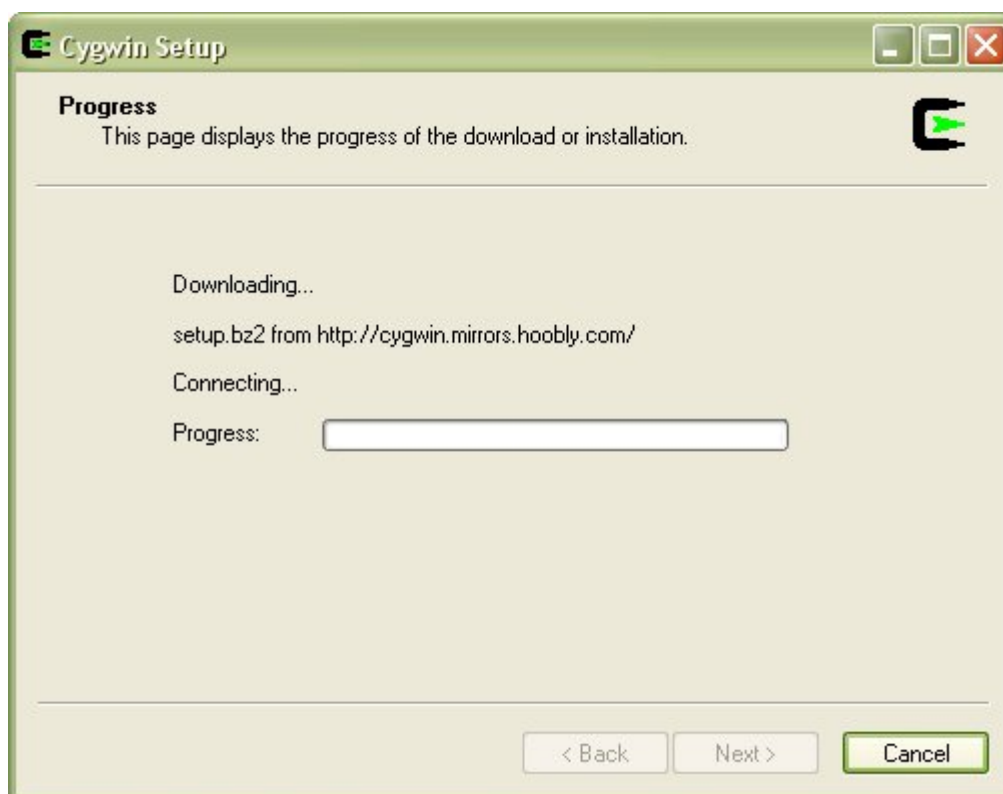
سپس پنجره بعدی باز می شود. در این پنجره یکی از لینکها را انتخاب می کنیم و دکمه next را فشار می دهیم.


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---



در این مرحله شکل زیر ظاهر می شود و لیست بسته های در دسترس برای نصب cygwin دانلود خواهد شد.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	--	---




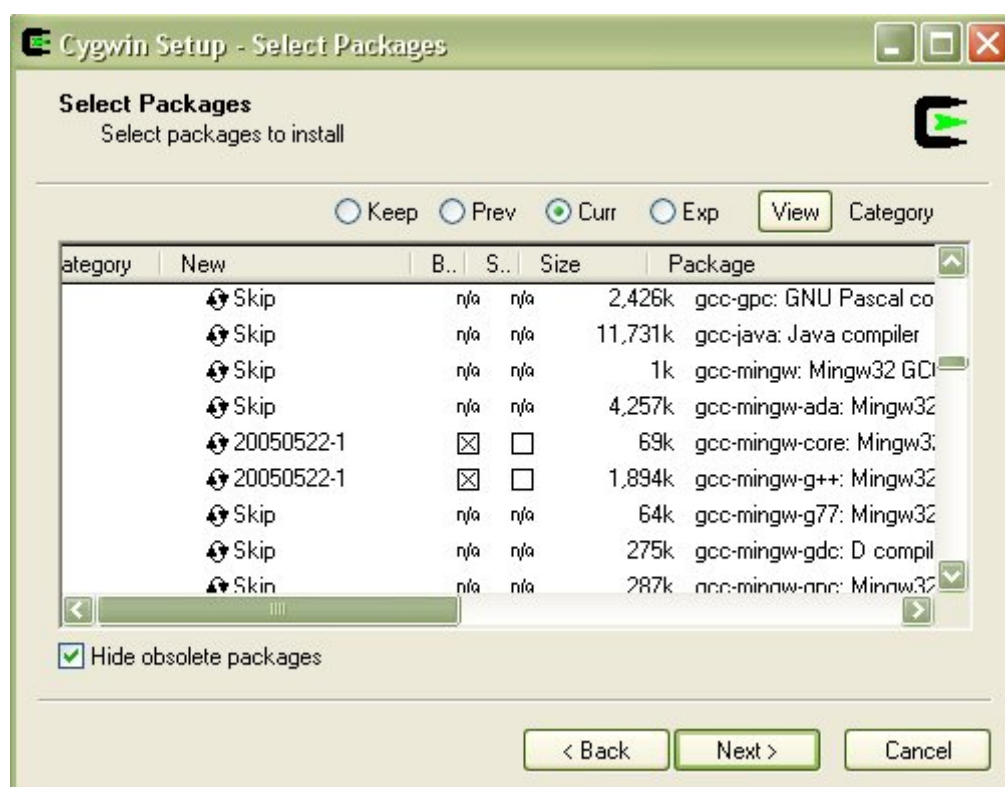
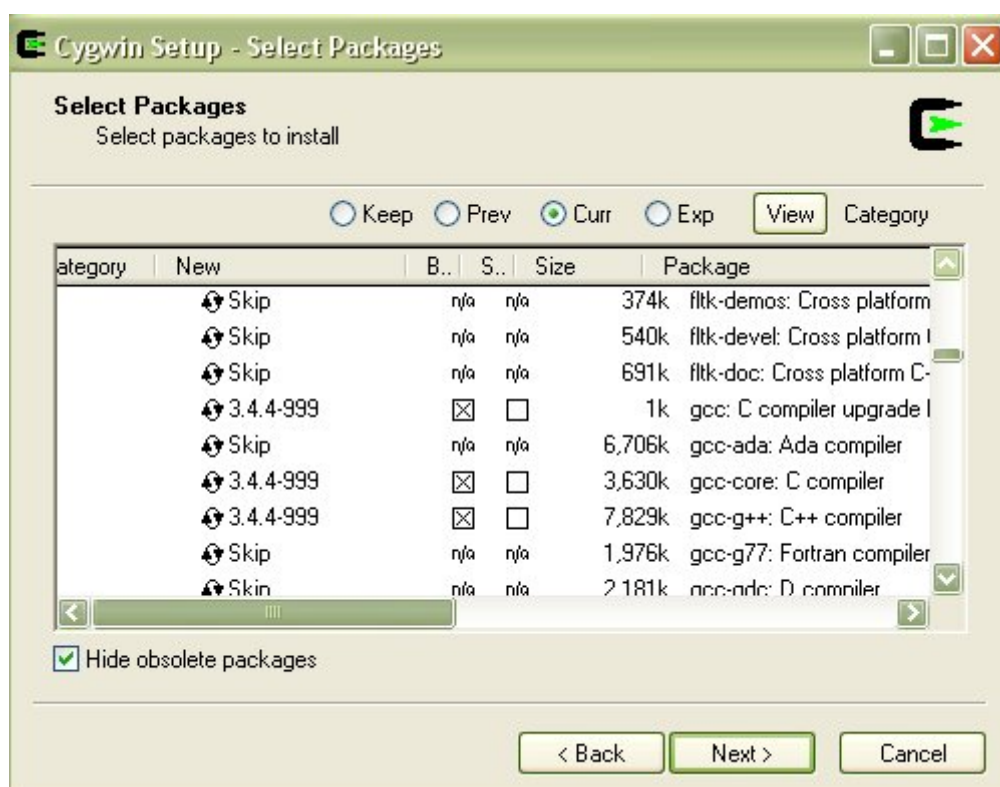
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---


در ادامه پنجره ای برای انتخاب بسته هایی که باید نصب شوند آورده شده تا برای دانلود انتخاب شوند..

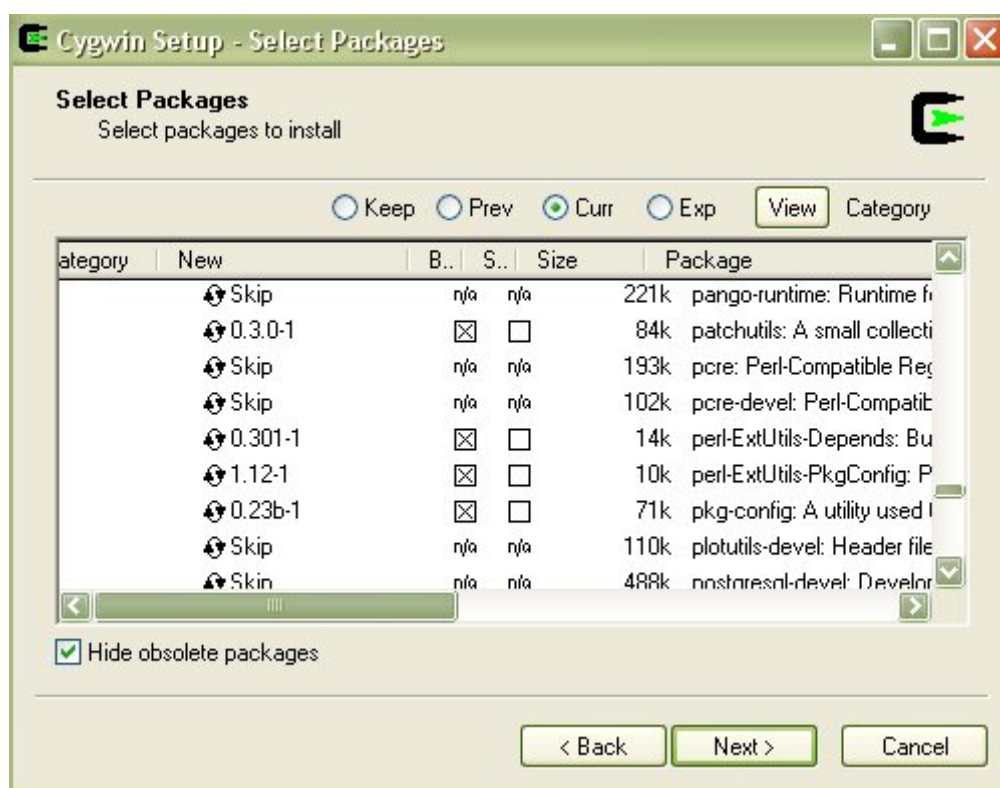


در این پنجره دسته devel را باز و در آن گزینه gcc را پیدا می کنیم.
روی گزینه های gcc, gcc-g++, patchutils, gcc-core و گزینه های مربوط به perl کلیک می کنیم تا برای نصب انتخاب گردند. سپس دسته net را باز کرده و گزینه inetutils را برای نصب انتخاب می کنیم.
در ادامه همین دسته گزینه های openssl و openssh را برای نصب انتخاب می کنیم و در ادامه گزینه make را نیز برای نصب انتخاب نموده و سپس در دسته X11 گزینه Xorg-x11-devel و Xorg-x11-devel را انتخاب می کنیم. در دسته editor گزینه vim را انتخاب می کنیم و سپس دکمه next را می زنیم.
مراحل در شکل های زیر نشان داده شده اند.


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
---	---	---

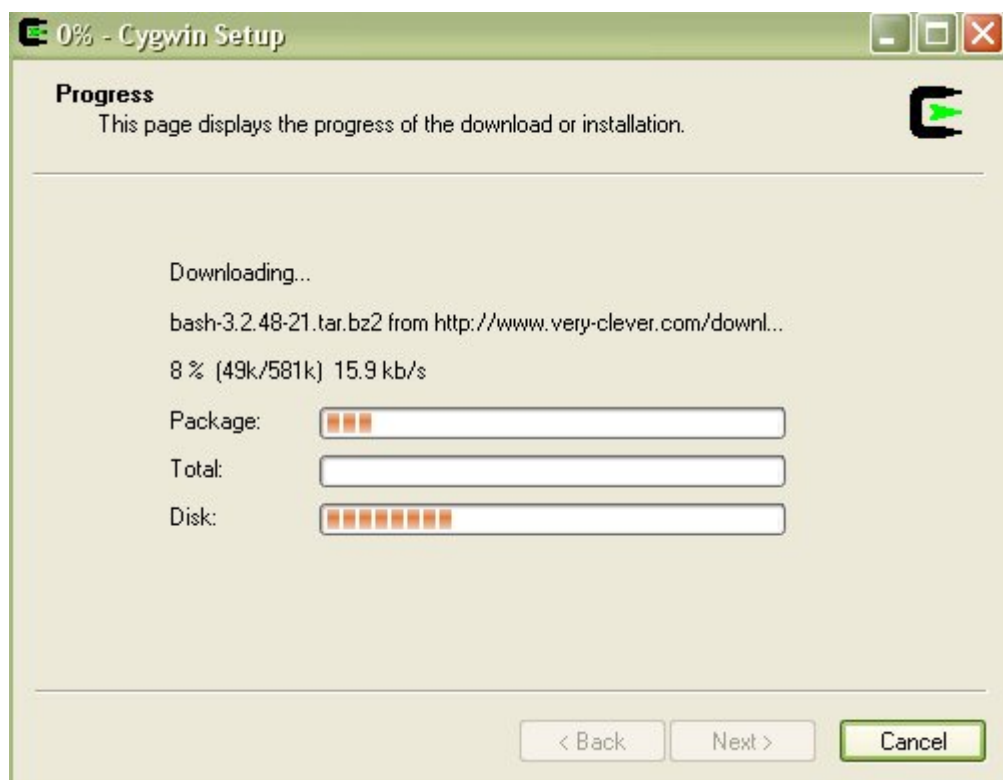


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---




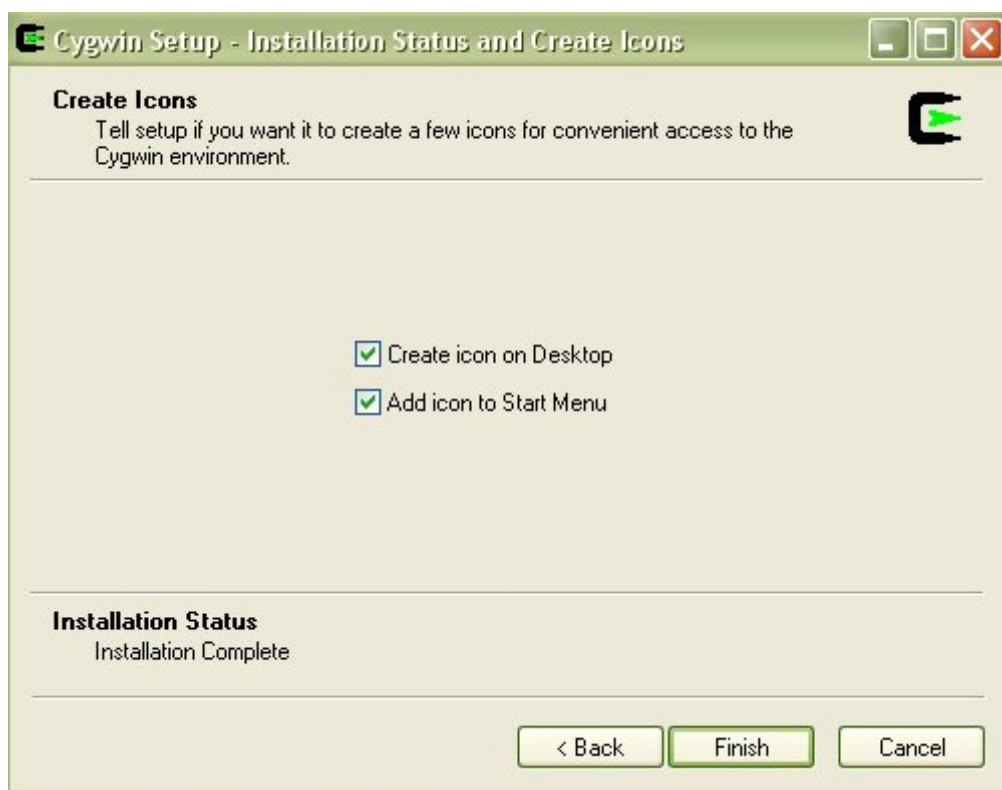
در ادامه پنجره ای نمایان می شود. در این پنجره بسته هایی برای نصب پیشنهاد می شوند که برای نصب کامل بسته هایی که در مراحل قبل انتخاب کرده ایم مورد نیاز است. بنابراین گزینه پایین پنجره را در حالت انتخاب شده رها کرده و دکمه next را میزنیم. در این مرحله برنامه set up شروع به دانلود کردن بسته های مورد نیاز برای نصب می شود.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---



این دانلود برای اینترنتی ۱۲۸ کیلو bps حدود ۴۵ دقیقه زمان خواهد برد. بعد از اینکه دانلود فایل ها به پایان رسید با اجرای برنامه setup ، نصب cygwin شروع می شود. پس از نصب برنامه setup از شما می خواهد در صورت تمایل اجازه دهید که در محیط desktop و در start menu ، shortcut هایی برای دسترسی سریع به bash shell ایجاد کند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---




دکمه finish را انتخاب میکنیم. در این مرحله نصب sygwin به پایان رسیده است.

برای اجرای برنامه cygwin روی آیکون  موجود در desktop کلیک می کنیم.



این پنجره محیط shellCygwin می باشد.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

Copying skeleton files.
These files are for the user to personalise
their cygwin experience.

These will never be overwritten.
'\\.bashrc' -> '/home/s//.bashrc'
'\\.bash_profile' -> '/home/s//.bash_profile'
'\\.inputrc' -> '/home/s//.inputrc'
s@w ~
$

```

این پنجره محیط shell linux را شبیه سازی می کند. فایل های ذخیره شده و فایل هایی که قرار است ذخیره شوند همگی درون پوشه ای قرار می گیرند که هنگام نصب انتخاب کرده ایم. اگر پیش فرض نصب را تغییر نداده باشیم این فایل ها و پوشه ها درون پوشه c:\cygwin ذخیره می شوند. برای نصب ns در محیط cygwin مراحل زیر را انجام می دهیم. در ابتدا فایل نصبی ns را درون پوشه c:\cygwin\home\s ذخیره می کنیم.

سپس به ترتیب دستورات زیر را اجرا می کنیم.


pwd

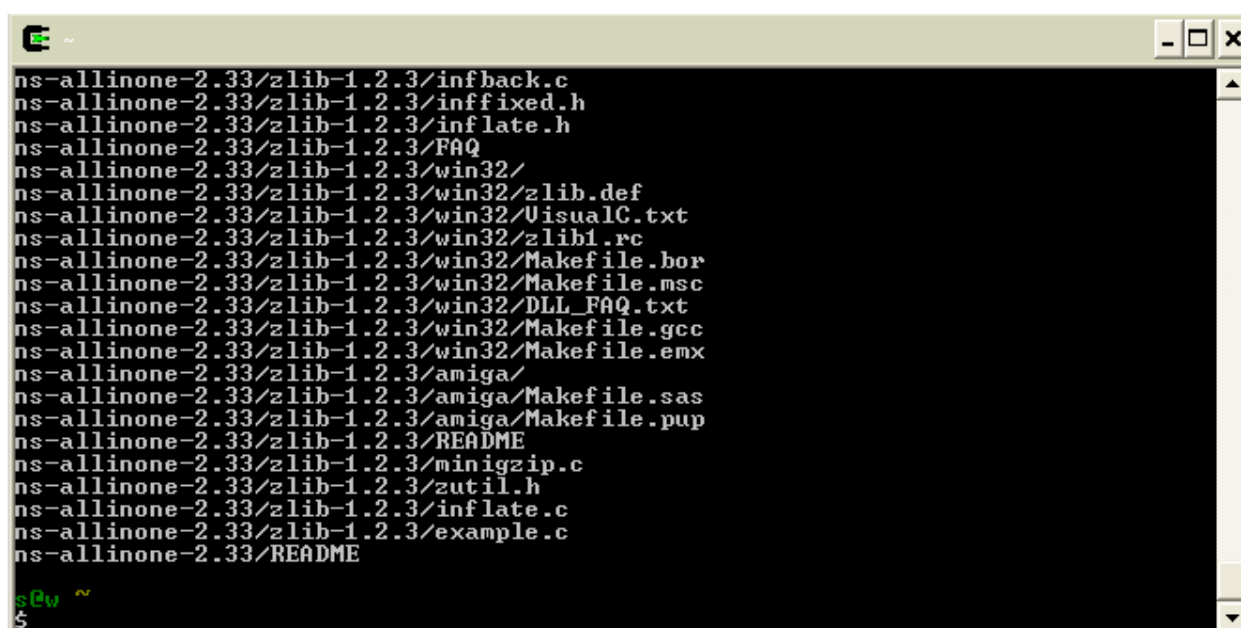
cd

Ls

Tar xzvf ns-allinone-2.33.tar.gz

آخرین فرمان به نام tar فایل فشرده شده ns را باز میکند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---





عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2

دانشگاه علم و صنعت ایران

مرکز آموزش های الکترونیکی

```

~/ns-allinone-2.33
ns-allinone-2.33/zlib-1.2.3/FAQ
ns-allinone-2.33/zlib-1.2.3/win32/
ns-allinone-2.33/zlib-1.2.3/win32/zlib.def
ns-allinone-2.33/zlib-1.2.3/win32/VisualC.txt
ns-allinone-2.33/zlib-1.2.3/win32/zlib1.rc
ns-allinone-2.33/zlib-1.2.3/win32/Makefile.bor
ns-allinone-2.33/zlib-1.2.3/win32/Makefile.msc
ns-allinone-2.33/zlib-1.2.3/win32/DLL_FAQ.txt
ns-allinone-2.33/zlib-1.2.3/win32/Makefile.gcc
ns-allinone-2.33/zlib-1.2.3/win32/Makefile.emx
ns-allinone-2.33/zlib-1.2.3/amiga/
ns-allinone-2.33/zlib-1.2.3/amiga/Makefile.sas
ns-allinone-2.33/zlib-1.2.3/amiga/Makefile.pup
ns-allinone-2.33/zlib-1.2.3/README
ns-allinone-2.33/zlib-1.2.3/minigzip.c
ns-allinone-2.33/zlib-1.2.3/zutil.h
ns-allinone-2.33/zlib-1.2.3/inflate.c
ns-allinone-2.33/zlib-1.2.3/example.c
ns-allinone-2.33/README

s@w ~
$ cd ns-allinone-2.33
s@w ~/ns-allinone-2.33
$

```

```

~/ns-allinone-2.33
rm -f test.gb sample.out test_io test_io.exe test_graph test_graph.exe test_flip
test_flip.exe test_sample test_sample.exe
echo "Congratulations --- the tests have all been passed."
Congratulations --- the tests have all been passed.
touch certified
=====
* Build GT-ITM
=====
gcc -I../include -L../lib -DFBSD -c -o itm.o itm.c
gcc -I../include -L../lib -DFBSD -c -o geog.o geog.c
gcc -I../include -L../lib -DFBSD -c -o ts.o ts.c
gcc -I../include -L../lib -DFBSD -c -o dfs.o dfs.c
gcc -I../include -L../lib -DFBSD -o ../bin/itm itm.o geog.o ts.o dfs.o -lm -lg
b
gcc -I../include -L../lib -DFBSD -c -o sgb2alt.o sgb2alt.c
gcc -I../include -L../lib -DFBSD -o ../bin/sgb2alt sgb2alt.o -lm -lgb
gcc -I../include -L../lib -DFBSD -c -o edriver.o edriver.c
gcc -I../include -L../lib -DFBSD -c -o eval.o eval.c
gcc -I../include -L../lib -DFBSD -o ../bin/edriver edriver.o eval.o -lm -lgb
gt-itm has been installed successfully.
gcc -I../include -L../lib -c -o sgb2comns.o sgb2comns.c
gcc -I../include -L../lib -o ../bin/sgb2comns sgb2comns.o -lm -lgb
gcc -I../include -L../lib -c -o sgb2hierns.o sgb2hierns.c
gcc -I../include -L../lib -o ../bin/sgb2hierns sgb2hierns.o -lm -lgb

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```
~/ns-allinone-2.33
TIME=1 -DHAUE_TZNAME=1 -DHAUE_GMTIME_R=1 -DHAUE_LOCALTIME_R=1 -DHAUE_TIMEZONE_U
AR=1 -DHAUE_ST_BLKSIZE=1 -DSTDC_HEADERS=1 -DNO_UNION_WAIT=1 -DHAUE_SIGNED_CHAR=1
-DHAUE_LANGINFO=1 -DHAUE_FTS=1 -DHAUE_SYS_IOCTL_H=1 -DTCL_SHLIB_EXT=""
/home/s/ns-allinone-2.33/tcl8.4.18/unix/./generic/tclUtf.c
gcc -c -O -pipe -DTCL_DBGX=-Wall -Wno-implicit-int -fno-strict-aliasing -I.
-I/home/s/ns-allinone-2.33/tcl8.4.18/unix/./generic -I/home/s/ns-allinone-2.33/
tcl8.4.18/unix -DNO_VALUES_H=1 -DHAUE_LIMITS_H=1 -DHAUE_UNISTD_H=1 -DHAUE_SYS_PA
RAM_H=1 -DSTATIC_BUILD=1 -DTCL_WIDE_INT_TYPE=long\ long -DHAUE_GETCWD=1 -DHAUE_O
PENDIR=1 -DHAUE_STRSTR=1 -DHAUE_STRTOL=1 -DHAUE_STRTOLL=1 -DHAUE_STROULL=1 -DHA
UE_TMPNAM=1 -DHAUE_WAITPID=1 -DUSE_TERMIOS=1 -DHAUE_SYS_TIME_H=1 -DTIME_WITH_SYS
TIME=1 -DHAUE_TZNAME=1 -DHAUE_GMTIME_R=1 -DHAUE_LOCALTIME_R=1 -DHAUE_TIMEZONE_U
AR=1 -DHAUE_ST_BLKSIZE=1 -DSTDC_HEADERS=1 -DNO_UNION_WAIT=1 -DHAUE_SIGNED_CHAR=1
-DHAUE_LANGINFO=1 -DHAUE_FTS=1 -DHAUE_SYS_IOCTL_H=1 -DTCL_SHLIB_EXT=""
/home/s/ns-allinone-2.33/tcl8.4.18/unix/./generic/tclUtil.c
gcc -c -O -pipe -DTCL_DBGX=-Wall -Wno-implicit-int -fno-strict-aliasing -I.
-I/home/s/ns-allinone-2.33/tcl8.4.18/unix/./generic -I/home/s/ns-allinone-2.33/
tcl8.4.18/unix -DNO_VALUES_H=1 -DHAUE_LIMITS_H=1 -DHAUE_UNISTD_H=1 -DHAUE_SYS_PA
RAM_H=1 -DSTATIC_BUILD=1 -DTCL_WIDE_INT_TYPE=long\ long -DHAUE_GETCWD=1 -DHAUE_O
PENDIR=1 -DHAUE_STRSTR=1 -DHAUE_STRTOL=1 -DHAUE_STRTOLL=1 -DHAUE_STROULL=1 -DHA
UE_TMPNAM=1 -DHAUE_WAITPID=1 -DUSE_TERMIOS=1 -DHAUE_SYS_TIME_H=1 -DTIME_WITH_SYS
TIME=1 -DHAUE_TZNAME=1 -DHAUE_GMTIME_R=1 -DHAUE_LOCALTIME_R=1 -DHAUE_TIMEZONE_U
AR=1 -DHAUE_ST_BLKSIZE=1 -DSTDC_HEADERS=1 -DNO_UNION_WAIT=1 -DHAUE_SIGNED_CHAR=1
-DHAUE_LANGINFO=1 -DHAUE_FTS=1 -DHAUE_SYS_IOCTL_H=1 -DTCL_SHLIB_EXT=""
/home/s/ns-allinone-2.33/tcl8.4.18/unix/./generic/tclVar.c
```

حال برای اجرا باید دستور install./ را فراخوانی کنیم. پس از اجرای این دستور برنامه ns در همین پوشه نصب می شود. پس از نصب دستوراتی داده می شود که باید آنها را اجرا کنیم.

حال دستورات مقابل را اجرا می کنیم تا فرمان های لازم برای تغییر مسیر اجرا شود.

LD_LIBRARY_PATH=/home/s/ns-allinone-2.33/otcl-1.12

LD_LIBRARY_PATH=/home/s/ns-allinone-2.33/lib


TCL_LIBRARY_PATH=/home/ns-allinone-2.33/tcl8.4.13/library

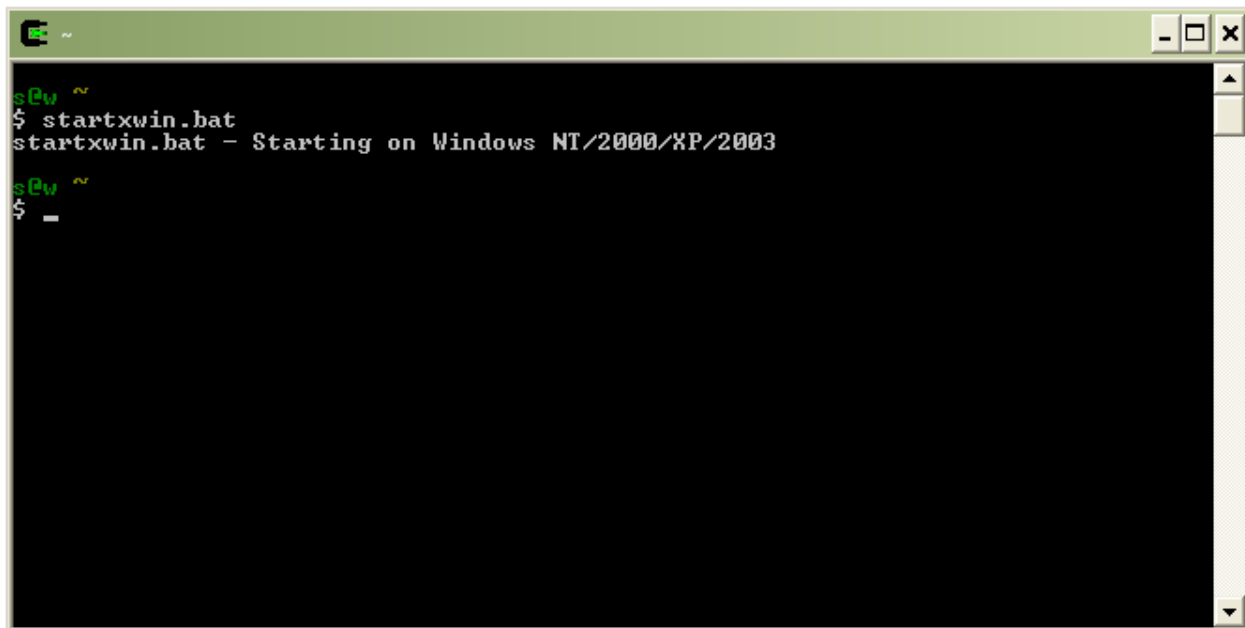
Export LD_LIBRARY_PATH

Export TCL_LIBRARY

توجه شود در صورتی که نیاز داریم نتایج شبیه سازی را در نرم افزار nam ببینیم باید دستور

startwxwin.bat را اجرا کنیم (شکل زیر)

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

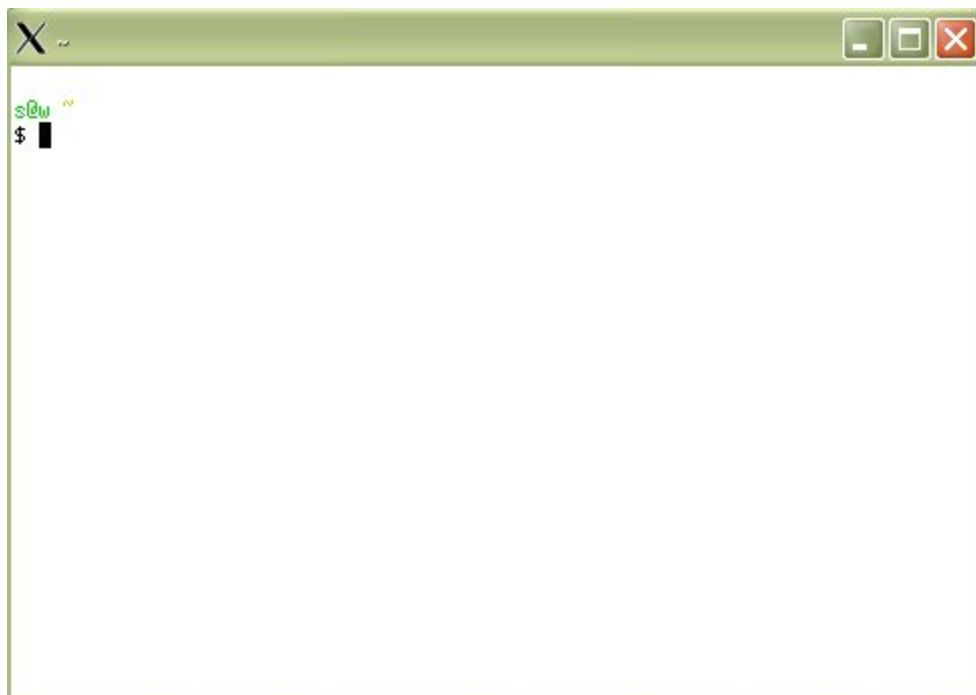


```

s@w ~
$ startxwin.bat
startxwin.bat - Starting on Windows NT/2000/XP/2003
s@w ~
$ -

```

در این حالت یک پنجره دیگر به شکل زیر باز خواهد شد.



۱-۳- شروع کار با N.S.

برای شروع کار کافی است برنامه موردنظر را که از دستورات موجود در N.S. و همچنین از دستورات TCL برای بیان توپولوژی شبکه و سایر ویژگی های شبکه موردنظر تشکیل شده است را در یک ویرایشگر

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

مثل emacs وارد کرده و آن را با پسوند tcl ذخیره نمود. سپس، برای اجرا کافی است دستور زیر را وارد نمایید:

نام فایل برنامه با پسوند ns tcl

خروجی برنامه را می توان به برنامه N.A.M داد، تا نتایج را به صورت Visual نمایش دهد و ما بتوانیم مراحل مختلف شبیه سازی را در فواصل زمانی مختلف مشاهده کنیم.

۱-۴- شبیه سازی N.S.

ویژگی های یک شبکه که باید در برنامه شبیه سازی لحاظ شود، عبارتند از:


- تعداد نودها
 - مشخصه هر نود
 - مشخصه لینک های اتصال دهنده نودها
 - نوع پروتکل های متصل به نودهای مبدا و مقصد
 - انواع ترافیک ارسالی
 - زمان شبیه سازی و زمانهای فعال شدن هر منبع ترافیکی
 - مشخصات فایل های مربوط به پایش ترافیک شبکه و استخراج پارامترهای کارایی شبکه.
- در اولین قدم در انجام شبیه سازی، باید یک شیء (object) از کلاس شبیه ساز ایجاد گردد. این امر با کمک دستور زیر قابل حصول است:

```
Set ns[new simulator]
```

بعد از دستور فوق با کمک دستورات ns و otcl توپولوژی شبکه که شامل نودها، لینک ها و عاملان شبکه (Agent) می باشند، ایجاد می شود. بعد از ایجاد نودها و عاملان شبکه، با کمک لینک ها نودهای شبکه به هم متصل می شوند و همچنین عاملان شبکه نیز به نودهای مربوطه اتصال می یابند. به عنوان مثال، پروتکل های مسیریابی دینامیکی، منابع ترافیکی و بسیاری از پروتکل های لایه ارسال نمونه ای از عاملان شبکه می باشند. بعد از ایجاد عاملان شبکه و اتصال آنها به نودها، می توان پارامترهای آنها را نیز تنظیم نمود. به عنوان مثال دو دستور زیر یک عامل ارسال از نوع TCL تعریف می نمایند و سپس پارامتر طول پنجره ارسال (window) که یک متغیر در پروتکل TCP است را به ۲۵ مقداردهی می کنند

```
Set tcp[new Agent/TCP]
$ tcp set window_25
```

برای زمانبندی رخدادها در N.S. با کمک دستور at می توان در هر لحظه دلخواه در طول زمان شبیه سازی روال های otcl را فعال نمود. بدین ترتیب امکان تعیین زمان شروع و پایان ارسال منابع ترافیکی، ایجاد خرابیهای موقت در لینک های شبکه در زمانهای خاص، پیکربندی مجدد توپولوژی شبکه و نظیر اینها فراهم می آید. به عنوان مثال دستورات زیر باعث می شوند که منبع ترافیکی از قبل تعیین شده \$cbr0 در زمان 0.5 شروع به ارسال ترافیک نماید و در زمان ۴/۵ متوقف گردد.

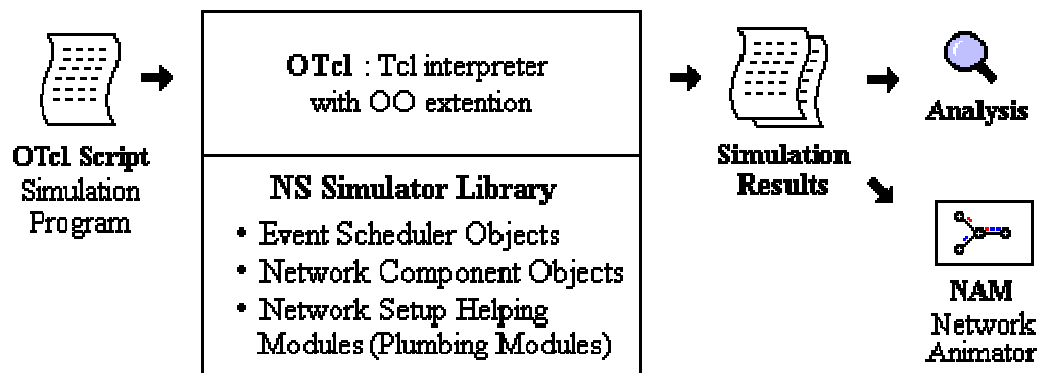
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

\$ns at 0.5 “\$cbr0 start”


\$ns at 4.5 “\$cbr0 stop”

با کمک دستور run شبیه سازی آغاز می شود و همچنین دستورات stop و exit باعث خاتمه عملیات شبیه سازی می شوند. آنچه در بالا آورده شد، توصیف اجمالی نحوه انجام عملیات شبیه سازی در N.S. است. در بخش های بعدی به توصیف کامل تر دستورات N.S. و TCL برای انجام شبیه سازیهای مختلف می پردازیم. هدف این گزارش آشنا ساختن خواننده با مفاهیم اولیه در ارتباط با شبیه سازی شبکه ها، چگونگی تنظیم پارامترهای موجود و چگونگی ایجاد اجزاء یک شبکه و غیره می باشد. در اینجا سعی می شود تا با عنوان مثالهایی نسبتاً ساده و از طریق توضیح آنها اطلاعات ضروری به خواننده داده شود. در ادامه خواننده قادر خواهد بود تا در صورت نیاز، با مراجعه به مراجع پیشرفته تر که آدرس آنها در انتهای گزارش آورده شده است، این راه را ادامه دهد.

شکل ۱ نمای کلی از اجزاء NS و طریقه کنار هم قرار گرفتن آنها را نشان می دهد. برای کار با NS شما باید یک برنامه به زبان OTCL بنویسید، سپس برای اجرای عملیات شبیه سازی باید عمل زمان بندی، ایجاد توپولوژی مناسب بین اجزای شبکه، تعریف نوع ترافیک موجود انجام پذیرد. در تمام مراحل فوق می توان از توابع کتابخانه ای موجود استفاده کرد.



شکل ۱: اجزای کلی ns

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

۱-۴-۱ OTCL زبان مورد استفاده کاربر

همانطور که قبلاً ذکر شد، نرم افزار NS بر اساس مفسر OTCL است که با کتابخانه های شبیه سازی شبکه کار می کند. دانستن اینکه در OTCL چگونه برنامه نویسی کنید برای کار با NS بسیار مفید خواهد بود. در این قسمت مثالی از یک برنامه TCL و OTCL به منظور آشنایی با مفاهیم مقدماتی برنامه نویسی در OTCL آورده شده است. این مثالها از پنجمین خودآموز NS آورده شده اند. در این قسمت فرض شده است که خواننده NS را نصب کرده است و با زبان های برنامه نویسی C و C++ آشنایی دارد.


مثال ۱، یک مثال کلی است تا نشان دهد که چگونه می توان در TCL یک پروسیجر تولید کرد و چگونه آنرا فرا خواند، همچنین چگونه می توان به متغیرها مقدار داد و یا یک حلقه درست کرد. از آنجائیکه OTCL نسخه شیءگرا (Object- Oriented) برنامه TCL است. بنابراین واضح است که تمام دستورات C و C++ است. برای شروع می توانید برنامه زیر را که به زبان TCL نوشته شده است در یک فایل بنویسید و آنرا با اسم ns ex-tcl.tcl ذخیره کنید. برای اجرای این برنامه در صفحه اجرایی عبارت مقابل را تایپ کنید ns ex-tcl.tcl دستور ns کار تفسیر OTCL را به وسیله NS آغاز می کند. اگر نسخه tcl8.0 یا بالاتر را در اختیار دارید می توانید همین کار را با تایپ عبارت ns ex-tcl.tcl انجام دهید.

```
# Writing a procedure called "test"
proc test () {
    set a 43
    set b 27
    set c [expr $a + $b]
    set d [expr [expr $a - $b] * $c]
    for {set k 0} {$k < 10} {incr k} {
        if {$k < 5} {
            puts "k < 5, pow = [expr pow($d, $k)]"
        } else {
            puts "k >= 5, mod = [expr $d % $k]"
        }
    }
}

# Calling the "test" procedure created above
test
```

مثال ۱: برنامه TCL ساده

در TCL، کلمه کلیدی proc برای تعریف یک پروسیجر بکار می رود. بعد از این کلمه کلیدی اسم پروسیجر و سپس آرگومانهای آن در براکت می آیند. برای مقداری به یک متغیر از کلمه کلیدی set استفاده می شود. عبارت [expr ...] به مفسر می گوید که مقدار عبارت داخل براکت که بعد از کلمه expr آمده است را محاسبه نماید. باید به این نکته توجه داشته باشید که برای در اختیار داشتن مقدار اختصاص یافته به یک متغیر از علامت \$ که به همراه اسم متغیر می آید استفاده کنید. برای چاپ مقادیر موردنظر

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

می توانید از کلمه کلیدی puts استفاده نمایید و مقادیر موردنظر را داخل " " قرار دهید. با اجرای این برنامه مقادیر زیر را در خروجی ملاحظه خواهید کرد.

```
K<5, pow=1.0
K<5, pow=1120.0
K<5, pow=1254400.0
K<5, pow=1404928000.0
K<5, pow=1573519360000.0
K<5, mod=0
K<5, mod=4
K<5, mod=0
K<5, mod=0
K<5, mod=4
```

مثال بعدی یک برنامه شیءگرا است که این بار در OTCL نوشته شده است. این مثال ساده چگونگی ایجاد یک شیء و استفاده از آن را در OTCL نشان می دهد. اگر به عنوان یک کاربر معمولی از NS استفاده می کنید، استفاده از یک شیء در برنامه ممکن است به ندرت پیش بیاید اما دانستن یک شیء در NS می تواند مفید باشد. شیء های NS که در برنامه های شبیه سازی مورد استفاده قرار می گیرند ممکن است به زبان C++ باشند اما بخاطر پیوندهای موجود بین OTCL و C++ در OTCL براحتی قابل استفاده هستند.


```
# add a member function call "greet"
Class mom
mom instproc greet {} {
    $self instvar age_
    puts "$age_ year old mom say:
    How are you doing?"
}

# Create a child class of "mom" called "kid"
# and override the member function 'greet'
Class kid -superclass mom
kid instproc greet {} {
    $self instvar age_
    puts "$age_ year old kid say:
    What's up, dude?"
}

# Create a mom and a kid object, set each age
set a [new mom]
$a set age_ 45
set b [new kid]
$b set age_ 15

# Calling member function "greet" of each object
$a greet
$b greet
```

مثال ۲: برنامه OTCL ساده

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

مثال ۲ یک برنامه OTCL است که دو کلاس شیء به اسم های mom و kid تعریف می کند که کلاس kid یک کلاس فرزند (child) از mom است و هر دو کلاس از یک تابع به نام greet استفاده می کنند. بعد از تعریف هر شیء، متغیر age برای کلاس mom به ۴۵ و برای کلاس kid به ۱۵ تنظیم می شود و سپس تابع greet برای هر کدام از شیء ها صدا زده می شود. کلمه کلیدی class برای تولید یک کلاس شیء و instproc برای تعریف تابع هر کلاس شیء مورد استفاده قرار می گیرند. کلمه کلیدی superclass برای تعریف خاصیت ارث بری بکار می رود و همانطور که در مثال قبل ملاحظه می کنید برای تعیین اینکه کلاس kid از نوع کلاس فرزند برای mom باشد از این کلمه استفاده شده است. در زبان C++ در تعریف توابع عضو در یک کلاس از اشاره گر this استفاده می شود و در اینجا \$self همان نقش را در OTCL ایفا می کند. Instvar چک می کند که آیا اسم متغیری که در ادامه می آید در کلاس مورد نظر تعریف شده است یا خیر. اگر این متغیر تعریف شده باشد، مورد استفاده قرار می گیرد در غیر این صورت یک متغیر جدید تعریف خواهد شد. برای تولید شیء، همانطور که در مثال ۲ مشاهده می کنید، از کلمه new استفاده می شود. اگر مثال ۲ را در برنامه ای با نام ex-otcl.tcl ذخیره کرده باشید با تایپ ns ex-otcl.tcl مقادیر زیر را مشاهده خواهید کرد.

45 year old mom say:

How are you doing?


15 year old kid say:

What' uper spective up, dude?

۱-۴-۲- یک مثال ساده از شبیه سازی

در این قسمت یک شبیه سازی در NS بیان می گردد و توضیح داده خواهد شد که هر خط چه کاری انجام می دهد. مثال ۳ یک برنامه OTCL است که شبکه ساده شکل ۲ را ایجاد می کند. برای اجرای شبیه سازی، برنامه زیر را در یک فایل به اسم ns-simple.tcl ذخیره کنید و عبارت ns ns-simple.tcl را در محیط اجرایی تایپ کنید. این شبکه از چهار گره به اسم های n0, n1, n2, n3 تشکیل شده است. یک لینک از نوع دو طرفه (duplex link) بین n0 و n2 و همچنین بین n1 و n3 با پهنای باند 2Mbps و تأخیر زمانی 10ms ایجاد می شود. بین n2 و n3 یک لینک دوطرفه با پهنای باند 1.7Mbps و تأخیر زمانی 20 ms قرار دارد. هر یک از گره ها از صف Drop Tail با حداکثر اندازه 10 استفاده می کنند.

یک عامل (agent) از نوع tcp به گروه n0 متصل می شود، همچنین یک اتصال برای عامل Sink متصل شده به n3 ایجاد می گردد. بر طبق پیش فرض حداکثر اندازه یک بسته که عامل tcp می تواند تولید کند یک کیلوبایت است. عامل sink وظیفه تولید و ارسال بسته های ACK برای فرستنده یعنی عامل tcp و نیز آزاد کردن بسته های دریافت شده را برعهده دارد. عامل udp متصل شده به n1 به عامل null متصل شده به n3 وصل است. عامل null فقط بسته های دریافت شده را آزاد می کند. ftp و cbr (Constant bit rate) تولید کننده های ترافیک هستند که به ترتیب به عوامل tcp و udp متصل می باشند.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

Cbr بسته های یک کیلوبایتی را با نرخ 1Mbps تولید می کند و طوری تنظیم شده است که در 0.1 ثانیه شروع بکار کرده و در ۴/۵ ثانیه متوقف می شود. ftp هم در ۱ ثانیه شروع و در ۴ ثانیه به کار خود پایان می دهد.

```
Create a simulator object
set ns [new Simulator]
```

```
#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red
```

```
#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf
```

```
#Define a 'finish' procedure
proc finish {} {
    global ns nf
    $ns flush-trace
    #Close the NAM trace file
    close $nf
    #Execute NAM on the trace file
    exec nam out.nam&
    exit 0
}
```

```
#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

```
#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
```

```
#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10
```

```
#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
```

```
#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5
```

```
#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
```

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```
$ns connect $tcp $sink
$tcp set fid_ 1
```

```
#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
```

```
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2
```

```
#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false
```

```
#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"
```


```
#Detach tcp and sink agents (not really necessary)
$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"
```

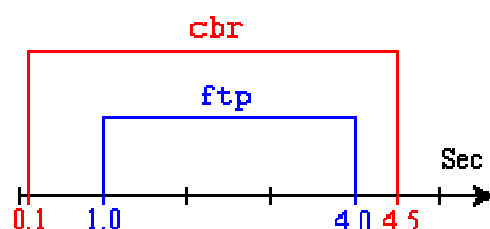
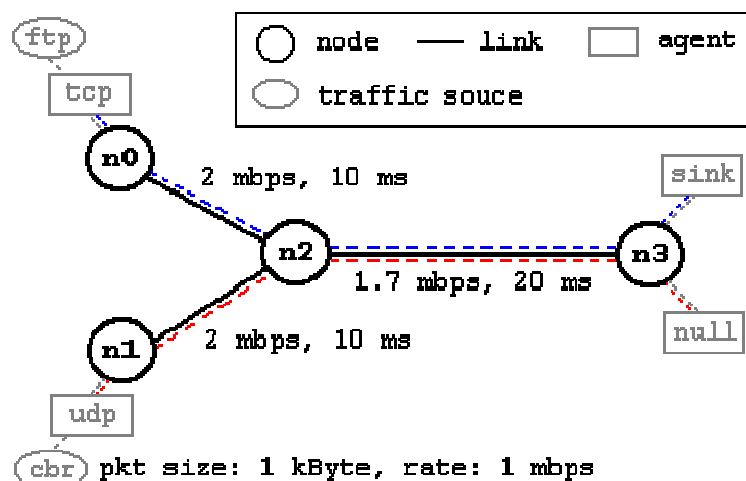
```
#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"
```

```
#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"
```

```
#Run the simulation
$ns run
```

مثال ۳: یک برنامه شبیه سازی ساده در NS

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---



شکل ۲: توپولوژی یک شبکه ساده

بطور کلی یک برنامه NS با تولید یک شیء شبیه ساز (Simulator Object) شروع بکار می کند.

Set ns [new simulator] -

باعث ایجاد یک شیء شبیه ساز می شود و آنرا به متغیر ns تخصیص می دهد. در اینجا برای متمایز شدن متغیرها و مقادیر از بقیه، آنها بصورت ایتالیک نمایش داده شده اند. این خط دستور کارهای زیر را انجام می دهد:

- فرمت بسته را مقداردهی اولیه می کند.
 - یک زمان بند (Scheduler) ایجاد می کند بصورت پیش فرض این زمان بند یک تقویم است.
 - فرمت آدرس پیش فرض را انتخاب می کند (فعلاً این قسمت را نادیده بگیرید).
- شیئی شبیه ساز دارای توابعی است که عملیات زیر را انجام می دهند.
- شیئی های ترکیبی مانند لینک ها و گره ها را تولید می کنند.
 - اجزای شیء های تولید شده نظیر عامل های tcp یا udp را وصل می کند.
 - پارامترهای اجزای شبکه که معمولاً برای شیء های ترکیبی مورد استفاده قرار می گیرند را تنظیم می کند.
 - بین عاملها مانند tcp و sink پیوند ایجاد می کند.
 - نحوه نمایش NAM را معین می کند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

○ غیره

بیشتر توابع برای تنظیم عملیات شبیه سازی و زمان بندی هستند هر چند که برخی از آنها برای نمایش NAM بکار می روند و می توانید این توابع را در قسمت ns-2/tcl/lib/ns-lib.tcl پیدا کنید.

\$ns color fid color:

برای تنظیم رنگ بسته های جریان مشخص شده بوسیله جریان id (fid) است. این تابع شیء شبیه ساز برای نمایش NAM بوده و بر روی شبیه ساز اصلی تأثیری ندارد. بعنوان مثال دستور \$ns color 1 Blue باعث می شود که جریان داده اول با رنگ آبی مشخص شود.

\$ns namtrace-all file-descriptor:

این تابع به شبیه ساز می گوید که ردیابی (trace) شبیه سازی را به شکل فرمت ورودی NAM ذخیره کند. همچنین اسم فایلی که نتایج ردیابی بعداً قرار است به وسیله دستور \$ns flush-trace در آن نوشته شود را می گیرد. تابع trace-all هم بطریق مشابه برای ذخیره سازی نتایج ردیابی با فرمت عمومی مورد استفاده قرار می گیرد.

Proc finish{ }:

بعد از شبیه سازی به وسیله دستور \$ns at s.o "finish" پروسیجر finish بعد از گذشت پنج ثانیه از شروع شبیه سازی فراخوانده می شود. در این تابع پروسه های بعد از شبیه سازی (post-simulation) مورد توجه قرار دارند.

Set n0 [\$ns node]:

این تابع گره ای را ایجاد می کند. در NS یک گره، یک شیء ترکیبی است که از آدرس و طبقه بندی کننده پورت (port classifier) که در بخش بعدی توضیح داده می شود، تشکیل شده است. کاربرها می توانند یک گره را از طریق ایجاد آدرس یا شیء طبقه بندی کننده پورت که به طور جداگانه ایجاد شده است و در ادامه به هم متصل می شوند، ایجاد نمایند. اما این تابع کار ایجاد یک گره را آسان کرده است. برای اینکه ببینید یک گره چگونه ساخته می شود می توانید به فایل های ns-2/tcl/lib/ns- و ns-2/tcl/lib/ns-lib.tcl مراجعه نمایید.

\$ns duplex-link node I node 2 bandwidth delay queue-type:

این دستور دو لینک ساده با تأخیر و پهنای باند معین ایجاد کرده و دو گره مشخص شده را به هم متصل می کند. در NS صف خروجی گره بعنوان بخشی از لینک محسوب می شود، بنابراین کاربرها باید نوع صف را در هنگام ایجاد لینکها مشخص کنند. مثلاً در مثال فوق صف از نوع Drop Tail تعریف شده است و اگر کاربر بخواهد از صف RED استفاده کند براحتی می تواند بجای کلمه Drop Tail کلمه RED را قرار دهد.

لینک هم مانند گره، یک شیء مرکب است و کاربرها می توانند اجزاء آن را به همراه گرهها به هم متصل کرده و آن را تولید کنند. برای مشاهده کدهای موجود می توانید به فایل های ns- ns2/tcl/lib/ns-lib.tcl یا ns-2/tcl/lib/ns-link.tcl مراجعه کنید. یک نکته قابل توجه این است که می توانید منابع خطا را در اجزاء

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

یک لینک وارد کنید و یک لینک با تلفات داشته باشید، برای آشنایی می توانید به راهنمای NS مراجعه کنید.

`$ns queue-limit node 1 node 2 number`

این دستور طول صف مربوط به دو لینک ساده، که گره های ۱ و ۲ را به هم متصل می کند، را با عدد داده شده مشخص می نماید. بعنوان مثال خط دستور `$queue-limit $n2 $n3 10` اندازه صف بین گره های `n2` و `n3` را برابر ۱۰ قرار می دهد.

`$ns duplex-link-op node 1 node 2 ...`

این دستور برای نمایش در NAM مورد استفاده قرار می گیرد. به وسیله این دستور می توانید موقعیت گره ها را نسبت به یکدیگر تعیین کنید. مثلاً خط فرمان:

`$ns duplex-link-op $n0 $n2 orient right-down`

تعیین می کند که گره `n2` نسبت به گره `n0` در موقعیت راست-پایین (جنوب شرقی) قرار بگیرد. با حذف خط دستور موقعیت گره ها و اجرای دوباره شبیه سازی می توانید تأثیر آنها را بهتر مشاهده نمایید. تا این جا مراحل اساسی شبیه سازی انجام شده است. در قسمت بعد باید عوامل ترافیک مانند TCP و UDP و منابع ترافیک مثل FTP و CBR که به ترتیب به گره ها و عوامل متصل می شوند، تنظیم شوند. بعد از شکل بندی شبکه، زمان بندی و شبیه سازی نهایی انجام می شوند و تنها چیزی که باقی مانده است اجرای شبیه سازی می باشد که به وسیله دستور `$ns run` انجام می شود.

Set tcp [new Agent/TCP]:


این خط چگونگی تولید یک عامل TCP را نشان می دهد. اما بطور کلی، کاربر می تواند هر عامل یا منبع ترافیک را بصورت زیر ایجاد کند. عوامل و منابع ترافیک در حقیقت اشیاء پایه ای و نه مرکب هستند که معمولاً در C++ ایجاد می شوند و سپس در OTCL مورد استفاده قرار می گیرند. بنابراین، هیچ تابع خصوصی وجود ندارد که آنها را تولید کند. برای ایجاد عوامل و منابع ترافیک، کاربر باید اسامی کلاس های این اشیاء را بداند (Application/FTP, Agent/TCPSink, Agent/TCP, و غیره). این گونه اطلاعات را می توانید در NS manual پیدا کنید اما برای راحتی کار می توانید به فایل `ns-2/tcl/libs/ns-default.tcl` مراجعه کنید. فایل مذکور حاوی مقادیر پارامترهای پیش فرض است و بنابراین می تواند کمک کند که چه اشیائی در NS موجود هستند و پارامترهای آنها چه هستند.

`$ns attach-agent node agent`

این دستور، عامل ایجاد شده را به گره مورد نظر وصل می کند. مثلاً `$ns attach-agent $n1 $udp` عامل `udp` را به گره `n1` وصل می کند.

`.$ns connect agent 1 agent 2`

بعد از اینکه دو عامل مانند Null یا Udp ایجاد شدند، باید یک ارتباط منطقی بین آنها ایجاد شود. بعنوان مثال دستور `$ns connect $Udp $Null` دو عامل ذکر شده را به هم متصل می کند. فرض کنید که تمام مراحل مربوط به پیکربندی (Configuration) شبکه انجام شده است، قدم بعدی نوشتن سناریوی

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

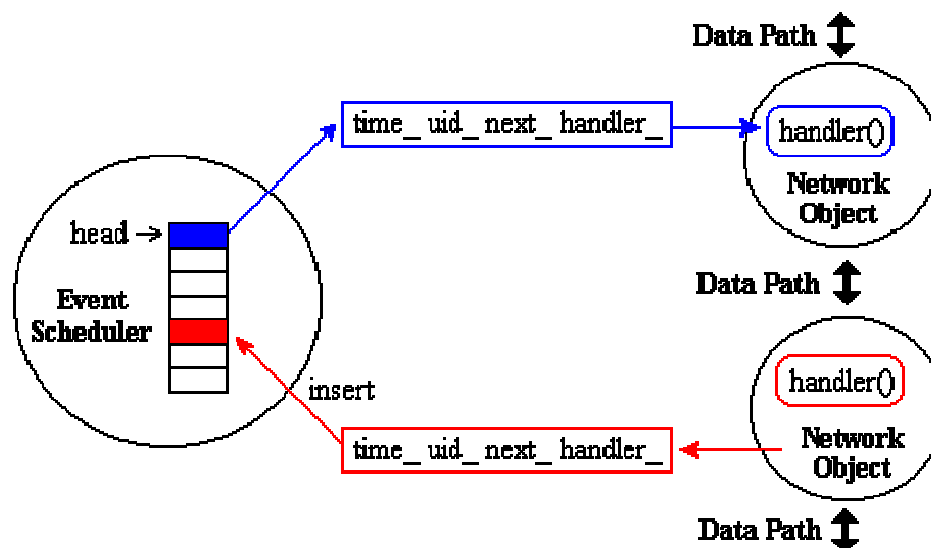
شبیه سازی یا عبارت دیگر زمان بندی شبیه سازی است. شبیه سازی دارای توابع زمان بندی گوناگون است که مهمترین آنها در زیر آمده است.

\$.ns at time "string"


این دستور باعث می شود که زمان بند (scheduler) اجرای عبارت مشخص شده در دستور را در زمان تعیین شده انجام دهد. بعنوان مثال دستور "\$cbr start" \$.ns at 0.1 سبب خواهد شد که زمان بند منبع ترافیک CBR را فعال ساخته و داده ها ارسال شوند. در NS معمولاً منبع ترافیک، داده واقعی را ارسال نمی کند، بلکه به عامل، اطلاع می دهد که مقداری داده برای ارسال دارد و عامل فقط می داند که چه میزان اطلاعات باید ارسال شود و بنابراین بسته هایی را تولید و آنها را ارسال می کند.

۱-۴-۳- زمان بند رخداد (Event Scheduler)

این بخش به بررسی زمان بندهای رخداد گسسته در NS اختصاص دارد. همانطور که در قسمت مقدمه ذکر شد، استفاده کننده های اصلی از زمان بندها، اجزا شبکه هستند که زمان رسیدگی به بسته (Packet-handling) را شبیه سازی می کنند و یا بخشهایی می باشند که به تایمر نیاز دارند. شکل ۳ هر یک از شیء های شبکه را که از زمان بند رخداد استفاده می کند نشان می دهد. باید توجه داشت که همان شیئی که یک رخداد را انتشار می دهد، رسیدگی به رخداد را بعداً در برنامه زمانی برعهده دارد. همچنین توجه کنید که مسیر داده ها بین اجزا شبکه با مسیر رخدادها متفاوت است. در حقیقت بسته ها از یک جزء شبکه به جزء دیگر به وسیله `Send (packet *P){trage_>recv(P)}` ارسال و به وسیله `recv (packet*, Handler *h=0)` دریافت می شوند.



شکل ۳: زمان بند رخداد گستر

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

NS از دو نوع مختلف زمان بند رخداد استفاده می کند. این دو نوع عبارتند از زمان بندهای زمان واقعی (real-time) و زمان غیرواقعی (non-real-time). برای یک زمان بند زمان غیرواقعی، سه اجرای Calender و Heap در دسترس هستند هرچند که تمام آنها تقریباً کار یکسانی را انجام می دهند. معمولاً Calender در زمان بند زمان غیرواقعی با همان مقادیر پیش فرض تنظیم می گردد. زمان بند زمان واقعی برای تقلید (emulation) بکار می رود تا به شبیه ساز این اجازه را بدهد که با شبکه واقعی ارتباط داشته باشد. در حال حاضر عمل تقلید در حال گسترش می باشد هرچند که نسخه آزمایشی آن هم اکنون در دسترس است. در زیر مثالی از انتخاب یک زمان بند رخداد خاص را مشاهده می کنید:

...

```
Set ns [new simulatr]
$ns use-scheduler Heap
```

...

یکی دیگر از کاربردهای زمان بند برای زمان بندی رخدادهای شبیه سازی است مثلاً چه موقع کاربرد FTP شروع بکار نماید یا چه زمانی کار شبیه سازی خاتمه یابد. زمان بند رخداد، خود دارای توابعی نظیر at time “string” می باشد که یک رخداد خاص که At Event نامیده می شود را در زمان معینی اجرا می کند. یک At Event در حقیقت یک کلاس فرزند از Event است که دارای یک متغیر اضافی برای نگهداری رشته داده شده به وسیله String می باشد. با این وجود مانند یک رخداد معمولی در زمان بند رخداد رفتار می کند. هنگامی که عمل شبیه سازی آغاز می شود و زمان At Event در زمان بند فرا می رسد آنگاه At Event به یک At Event handler داده می شود که این رسیدگی کننده (handler) یک بار ایجاد می شود و برای تمام At Event ها مورد استفاده قرار می گیرد و دستور OTCL که به وسیله قسمت Sting مشخص شده است، اجرا می شود. در ادامه زمان بند رخداد شبیه سازی به مثال قبل اضافه شده است.

...

```
Set ns [new simulatr]
$ns use-scheduler Heap
$ns at 300.s “complete-sim”
```

...


```
Proc complete-sim { } {
```

...

```
}
```

ممکن است به این موضوع توجه کرده باشید که at time “string” یک تابع عضو شیئی شبیه ساز ns[new simulation] است. اما بخاطر داشته باشید که این شیئی شبیه ساز فقط بعنوان یک واسط (interface) کاربر عمل می کند و در واقع توابع عضو اشیاء شبکه یا شیئی زمان بند را صدا می زند تا کار واقعی را آنها انجام دهند. در ادامه لیستی از توابع عضو شیئی شبیه ساز که با توابع عضو زمان بند در ارتباط هستند به همراه توضیح مختصری درباره آنها آمده است.

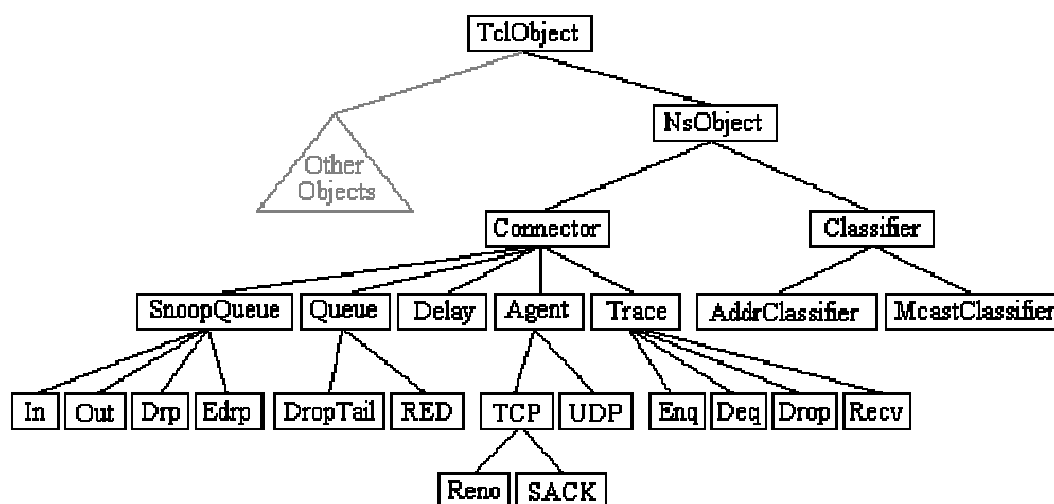
```
Simulator instproc now          # return scheduler's notion of current
                                time
Simulator instproc at args      # scheduler execution of code specified
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

	time
Simulator instproc at-now args	# scheduler execution of code at now
Simulator instproc after n args	# scheduler execution of code after n secs
Simulator instproc run arge	# start scheduler
Simulator instproc halt	# stop (pause) scheduler


۱-۴-۴- اجزای شبکه Network components

این بخش درباره اجزای NS که معمولاً اجزای ترکیبی شبکه هستند صحبت می کند. شکل ۴ قسمتی از سلسله مراتب کلاسهای NS را نشان می دهد که می تواند برای درک اجزای اصلی شبکه مفید باشد. اگر می خواهید سلسله مراتب کامل کلاسهای موجود در NS را ببینید می توانید به آدرس زیر مراجعه کنید:
<http://www.sop.inria.fr/rodeo/personnel/Antoine.Clerget/ns>.



شکل ۴: بخشی از سلسله مراتب کلاس

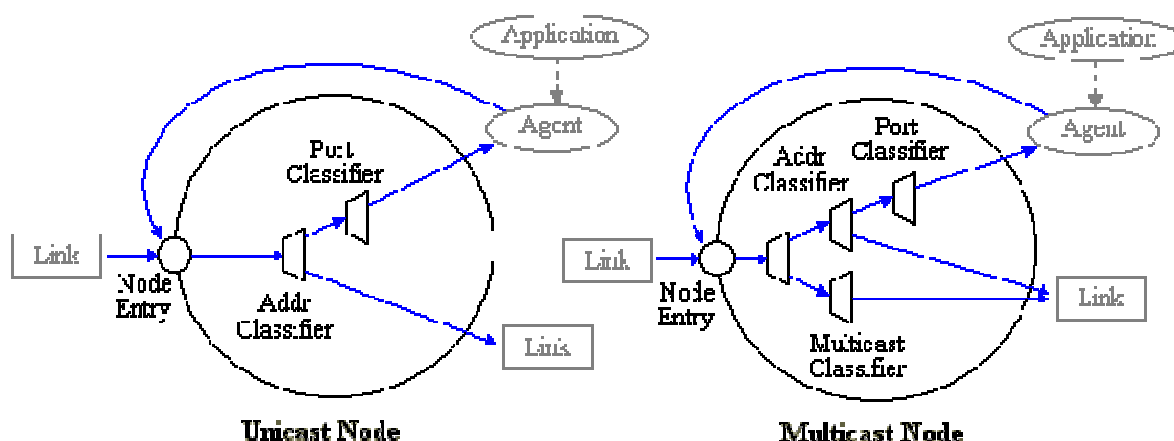
ریشه اصلی این سلسله مراتب، کلاس OTCL Object است که یک فوق کلاس (Superclass) از تمام اشیاء کتابخانه OTCL از قبیل زمان بند، اجزای شبکه، تایمرها و بقیه اشیاء مرتبط با NAM می باشد. کلاس NS object بعنوان یک کلاس از اجداد کلاس OTCL object است که خود بصورت فوق کلاس از تمامی اجزای اصلی شبکه مثل گره ها و لینک ها که کار رسیدگی و جابجایی بسته ها را بر عهده دارند، است. اجزای اصلی شبکه هم به نوبه خود به دو زیر کلاس طبقه بندی کننده (Classifier) و رابط (Connector) تقسیم می شوند. این تقسیم بندی بر اساس تعداد مسیرهای ممکن از داده خروجی انجام می گیرد. آن اجزایی از شبکه که فقط یک مسیر داده خروجی را دارند در کلاس رابط و آنهایی که دارای چندین مسیر داده خروجی هستند در کلاس طبقه بندی کننده قرار می گیرند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۴-۵- گره و مسیریابی

گره یک شیئی ترکیبی است که از دو بخش ورودی و طبقه بندی کننده تشکیل شده است (شکل ۵). دو نوع گره در NS وجود دارد. نوع اول تک پخششی (unicast) است و دارای طبقه بندی کننده آدرس، که عمل مسیریابی تک پخششی را انجام می دهد، و طبقه بندی کننده پورت می باشد. نوع دوم گره چندپخششی است که علاوه بر اجزای گره تک پخششی، دارای طبقه بندی کننده، که بسته های چندپخششی را از بسته های تک پخششی جدا می کند، و نیز طبقه بندی کننده چندپخششی است که عمل مسیریابی چندپخششی را بر عهده دارد. در NS گره ها بصورت پیش فرض از نوع تک پخششی انتخاب می شوند و برای ایجاد گره های چندپخششی، کاربر باید بصورت صریح در متن برنامه OTCL ورودی اعلام کند که گره ها از نوع چندپخششی هستند. این اعلام باید درست بعد از ایجاد شیئی زمان بند صورت پذیرد. بعد از اینکه نوع گره مشخص شد، کاربر می تواند یک قرارداد مسیریابی مشخص را بجای قرارداد مسیریابی پیش فرض انتخاب کند. برای تعریف گره بصورت تک پخششی از دستور زیر استفاده کنید:

```
$ns rtproto type
```



شکل ۵: گره چندپخششی و تک پخششی

که در قسمت type می توانید از انواع زیر برای گره تک پخششی استفاده کنید:

Static, Session, DB, Cost, multi-path

برای تعریف گره بصورت چندپخششی از دستور زیر استفاده کنید:

```
$ns multicast
```

```
$ns mrtproto type
```

در قسمت type می توان یکی از انواع زیر را انتخاب کنید:

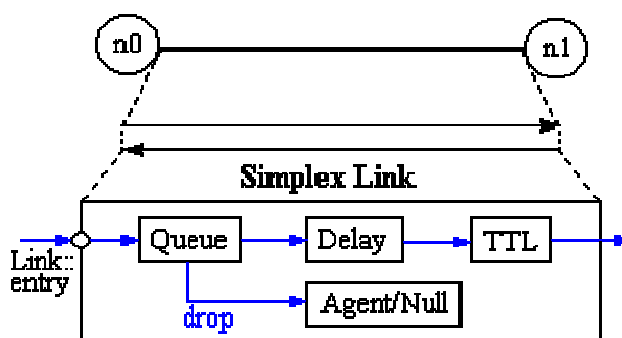
CtrMcast, DB, St, BST

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

باید به این نکته دقت کنید که دستور \$ns multicast باید دقیقاً [new Scheduler] set \$ns باشد. برای اطلاعات بیشتر می توانید به NS Manual در آدرس زیر مراجعه کنید. در آنجا چند فصل درباره مسیریابی چندپخشی و تک پخشی وجود دارد.

<http://www.isi.edu/nsnam/ns/ns-documentation.html>


مانند گره ها، لینک ها هم جزء اجزای ترکیبی در NS می باشند. وقتی که کاربر یک لینک از نوع دوطرفه به وسیله توابع عضو ایجاد کرد، آنگاه دو لینک ساده یک طرفه در هر دو جهت مطابق شکل ۶ ایجاد می شود. یک نکته قابل توجه این است که صف خروجی از گره در حقیقت بعنوان قسمتی از لینک ساده عمل می کند. بسته هایی که در صف قرار می گیرند به قسمت تأخیر (Delay) که عمل شبیه سازی تأخیر لینک را برعهده دارد، فرستاده می شوند و بسته هایی که از صف بیرون انداخته می شوند به قسمت Null رفته و آنجا آزاد می شوند. در انتها قسمت TTL (time to live) پارامترهای زمان حیات هر بسته دریافت شده را محاسبه می کند و قسمت TTL بسته ها را با مقدار جدیدی تغییر می دهد.



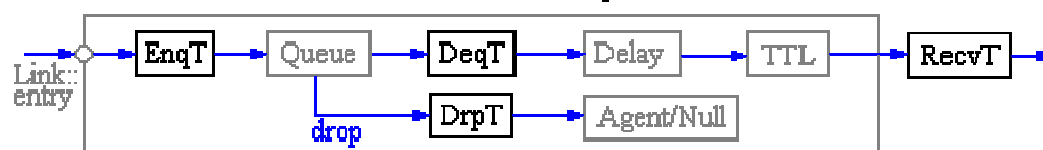
شکل ۶: لینک ساده

۱-۴-۶- ردیابی (Tracing)

در NS، فعالیت های شبکه در یک لینک ساده ردیابی می شوند. اگر به شبیه ساز از طریق دستورهای \$ns trace-all file و یا \$ns namtrace-all file ردیابی فعالیت های شبکه ابلاغ شود، آنگاه لینک ها بعد از اجرای دستورهای فوق قسمتهایی را داخل لینک مطابق شکل ۷ وارد می کنند. کاربرها هم می توانند بطور مخصوص عمل ردیابی بین گره های مبدأ (src) و مقصد (dst) را با اجرای دستور create-trace {type file src dst} انجام دهند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

Link with Trace Objects



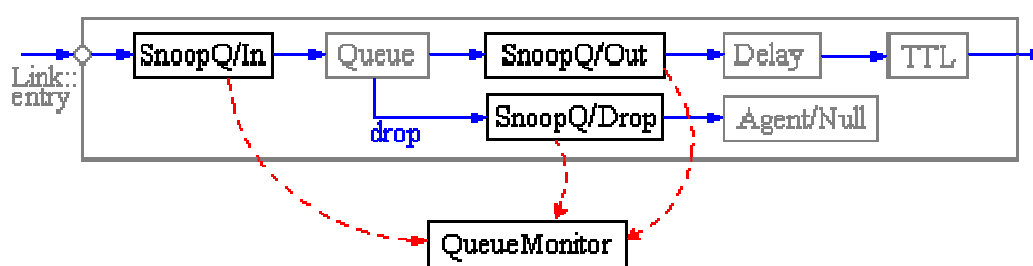
شکل ۷: وارد کردن شیئی های ردیابی

هنگامی که هر یک از قسمت های اضافه شده برای ردیابی یعنی EnqT, DeqT, DrpT و RcvT یک بسته را دریافت کردند، آنگاه توضیحی را در یک فایل ردیابی بدون تلف کردن زمان شبیه سازی می نویسند و بسته را به قسمت بعدی تحویل می دهند. قالب ردیابی در قسمت آنالیز عمومی مسئله مورد بررسی قرار خواهد گرفت.

۲-۹-۱ پایش صف (Queue Monitor)

اصولاً عمل ردیابی برای ذخیره کردن زمان رسیدن بسته به قسمت های ردیابی طراحی شده است. هرچند یک کاربر می تواند اطلاعات کافی را از ردیابی بدست آورد اما ممکن است اتفاقات انجام شده داخل یک صف خروجی خاص برایش جالب باشد. بعنوان مثال یک کاربر به رفتار صف RED علاقه مند بوده و می خواهد اندازه متوسط صف و اندازه صف جاری خاصی از نوع RED را اندازه گیری کند یا به عبارت ساده تر نیاز به مشاهده صف دارد. مشاهده صف را می توان با استفاده از شیئی های مشاهده صف و شیئی های جستجوکننده (Snoop) صف که در شکل ۸ نشان داده شده است، انجام داد.

Link with Snoop Queue Objects




شکل ۸: مشاهده صف

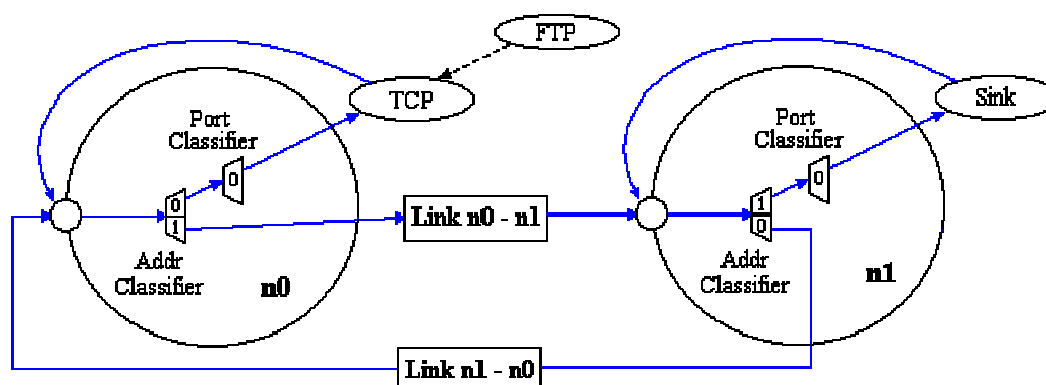
هنگامی که یک بسته دریافت می شود، قسمت جستجو کننده صف به قسمت نمایش صف این رخداد را اعلام می کند و قسمت نمایش صف با این اطلاعات صف را نشان می دهد. نمایش صف RED در یک مثال نشان داده خواهد شد. توجه کنید که قسمت های جستجوکننده صف می توانند بصورت موازی با قسمت های ردیابی قرار گیرند، اگرچه در شکل ۸ نشان داده نشده است.

۲-۹-۲ مثالی از جریان بسته

تا اینجا دو قسمت مهم شبکه یعنی گره و لینک مورد بررسی قرار گرفته اند. شکل ۹ مثالی از شبیه سازی حرکت بسته را داخل یک شبکه شامل دو گره n_0 و n_1 با آدرسهای 0 و 1 نشان می دهد. عامل TCP که به

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---


گره $n0$ متصل است، از طریق پورت صفر با قسمت TCP sink که به پورت صفر گره $n1$ وصل است، ارتباط دارد. یک کاربرد FTP یا در حقیقت منبع ترافیک به عامل TCP متصل شده است و درخواست ارسال مقداری داده را دارد.

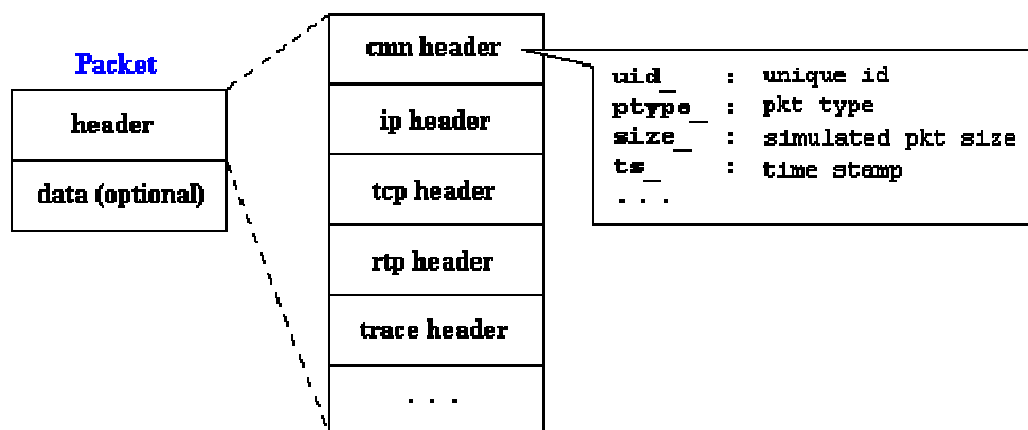


شکل ۹: مثالی از حرکت بسته

۱-۴-۷- بسته

در NS یک بسته متشکل از یک پشته (stack) از سرآمدها و قسمت داده اختیاری می باشد. همانطور که در مثال ساده از شبیه سازی بطور خلاصه گفته شد، فرمت سرآمد بسته در هنگام ایجاد شیئی شبیه ساز، مقداردهی اولیه می شود. پشته، شامل سرآمدهای cmm که مورد نیاز همه اشیاء می باشد، سرآمد IP، سرآمد TCP، سرآمد RTR (از سرآمد RTP استفاده می کند) و بالاخره سرآمد trace است. برای هر یک از سرآمدها در پشته، میزان افست در محلی ذخیره می شود. به این معنی که هرگاه یک سرآمد معین مورد استفاده قرار گیرد، هنگام تخصیص بسته به وسیله یک عامل، پشته ای از سرآمدهای ذخیره شده تولید خواهد شد و بنابراین یک شیئی شبکه می تواند به هر یک از سرآمدهای داخل پشته دسترسی داشته باشد. این دسترسی، از طریق پردازش های لازم بر روی مقدار افست مربوطه مقدور خواهد بود.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---




شکل ۱۰: فرمت بسته در NS

معمولاً یک بسته فقط شامل پشته ای از سرآمدها است و اشاره گر به فضای داده مقدار Null را دارد. اگر چه یک بسته می تواند داده واقعی را از یک کاربر، با تخصیص فضای داده حمل کند، اما تعداد کاربردها و عاملهایی که از این روش استفاده می کنند بسیار کم است، چرا که حمل داده در یک شبیه سازی بصورت زمان غیرواقعی، بی معنی است. اما اگر شما بخواهید یک کاربرد با کاربرد دیگر در شبکه صحبت کند، می توانید از روش حمل داده، با کمی تغییر در اجرای عامل مورد نظر استفاده نمایید. روش دیگر، ایجاد یک سرآمد جدید برای کاربرد و تغییر در عامل مورد نظر برای نوشتن داده دریافت شده از کاربرد در آن سرآمد جدید می باشد. روش دوم را در قسمت اضافه کردن یک کاربرد و عامل جدید، تحت یک مثال خواهید دید.

۸-۴-۱- مثال تجزیه و تحلیل ردیابی

این قسمت مثالی از تجزیه و تحلیل مربوط به ردیابی را نشان می دهد. مثال ۴ یک برنامه OTCL است که شبیه مثال قسمت (مثالی از شبیه سازی ساده) می باشد، اما فقط چند خط برای باز کردن یک فایل ردیابی و نوشتن نتایج ردیابی در آن فایل به برنامه اضافه شده است. توپولوژی شبکه مانند شکل ۴ در قسمت مثالی از شبیه سازی ساده می باشد. با اجرای برنامه بالا یک فایل ردیابی NAM ایجاد می شود که بعنوان ورودی برای برنامه NAM مورد استفاده قرار می گیرد، همچنین یک فایل ردیابی به اسم out.tr نیز تولید می شود که برای تحلیل شبیه سازی از آن استفاده خواهیم کرد. شکل ۱۱ فرمت ردیابی و مثالی از داده ردیابی فایل out.tr را نشان می دهد. هر خط مربوط به ردیابی با یکی از توصیف گرهای +, -, d, r شروع می شود. سپس زمان شبیه سازی رخداد بر حسب ثانیه می آید و در ادامه قسمت از گره تا گره قرار دارد و لینکی را که قرار است رخداد بر روی آن اتفاق بیفتد را مشخص می کند.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

* * *

```
#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Open the Trace file
set tf [open out.tr w]
$ns trace-all $tf

#Define a 'finish' procedure
proc finish {} {
    global ns nf tf
    $ns flush-trace
    #Close the NAM trace file
    close $nf
    #Close the Trace file
    close $tf
    #Execute NAM on the trace file
    exec nam out.nam &
    exit 0
}

* * *
```

مثال ۴: فعال سازی ردیابی در مورد یک برنامه ساده NS


event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id
-------	------	-----------	---------	----------	----------	-------	-----	----------	----------	---------	--------

```
r : receive (at to_node)
+ : enqueue (at queue)          src_addr : node.port (3.0)
- : dequeue (at queue)          dst_addr : node.port (0.0)
d : drop (at queue)

r 1.3556 3 2 ack 40 ----- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ----- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ----- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ----- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ----- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ----- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ----- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ----- 2 1.0 3.1 157 207
```

شکل ۱۱: مثالی از فرمت ردیابی

شکل ۹ در قسمت (اجزای شبکه) نشان می دهد که کجای لینک، هر نوع از رخدادها را دنبال می کند. توصیف گر r به معنی دریافت در گره مشخص شده در قسمت to-node است. توصیف گر d هم به معنی انداختن (drop) در صف می باشد. توصیف گرهای + و - هم به ترتیب برای انجام عملیات enqueue و dequeue در صف هستند. قسمت بعدی در خط ردیابی که قبل از بخش پرچم (flag) قرار دارد، اندازه و نوع بسته هستند که برحسب بایت بیان می شوند.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

در شکل ۱۱ پرچم ها با علام "....." مشخص شده اند که بیان می کنند هیچ پرچمی هنوز تنظیم نشده است. معمولاً NS فقط با ECN (Explicit Congestion Notification) کار می کند و بقیه بیت ها را استفاده نمی کند. قسمت بعدی شناسایی جریان (flow identification) یا بطور خلاصه fid است که کاربر می تواند آنرا در برنامه OTCL برای هر یک از جریانها تنظیم کند. اگر چه قسمت fid ممکن است در شبیه سازی استفاده نشود اما کاربرها می توانند از این قسمت برای مقاصد تجزیه و تحلیل استفاده نمایند. یکی دیگر از موارد استفاده از قسمت fid وقتی است که در نمایش برنامه NAM از رنگ ها استفاده می شود. دو قسمت دیگر بعد از fid آدرسهای مبدا و مقصد هستند که به فرم node.port می باشند. بعنوان مثال آدرس گره مبدا و مقصد می تواند بصورت زیر باشد:

Src-addr: node.port (3.0)

Dst-addr: node.port (0.0)

قسمت بعدی شماره سکانس بسته مربوط به قرارداد لایه شبکه را نشان می دهد. توجه داشته باشید که اگر چه UDP از شماره سکانس استفاده نمی کند اما NS شماره سکانس بسته را نگهداری می کند تا برای مقاصد تجزیه و تحلیل مورد استفاده قرار دهد. قسمت آخر هم مربوط به id بسته می باشد که یک مقدار منحصر به فرد است و برای هر بسته متفاوت می باشد.

حال که داده های مربوط به ردیابی را در اختیار داریم، تنها کار لازم تبدیل قسمتی از داده های مورد علاقه به اطلاعات قابل درک و سپس تجزیه و تحلیل آنها می باشد. در زیر مثالی از تبدیل داده ای که مقدار کمی دارد، آورده شده است. در این برنامه از دستور column استفاده شده است که ستونهایی از داده را انتخاب می کند. برای اینکه این برنامه روی کامپیوتر شما اجرا شود باید ابتدا column را بصورت قابل اجرا در آورید. برای این کار باید دستور `chmod 755 column` را در خط فرمان اجرا نمایید. دستور `awk` هم وظیفه محاسبه لرزش (jitter) ترافیک CBR که در گره n3 دریافت می شود را بر اساس داده فایل `out.tr` برعهده دارد. این دستور همچنین نتایج را در فایل `jitter.txt` ذخیره می کند.

```
Cat out.tr |grep "2 3 cbr" |grep ^r | column 1 10 |awk '{dif=2-old2;
```


```
If (dif==0) dif=1;
```

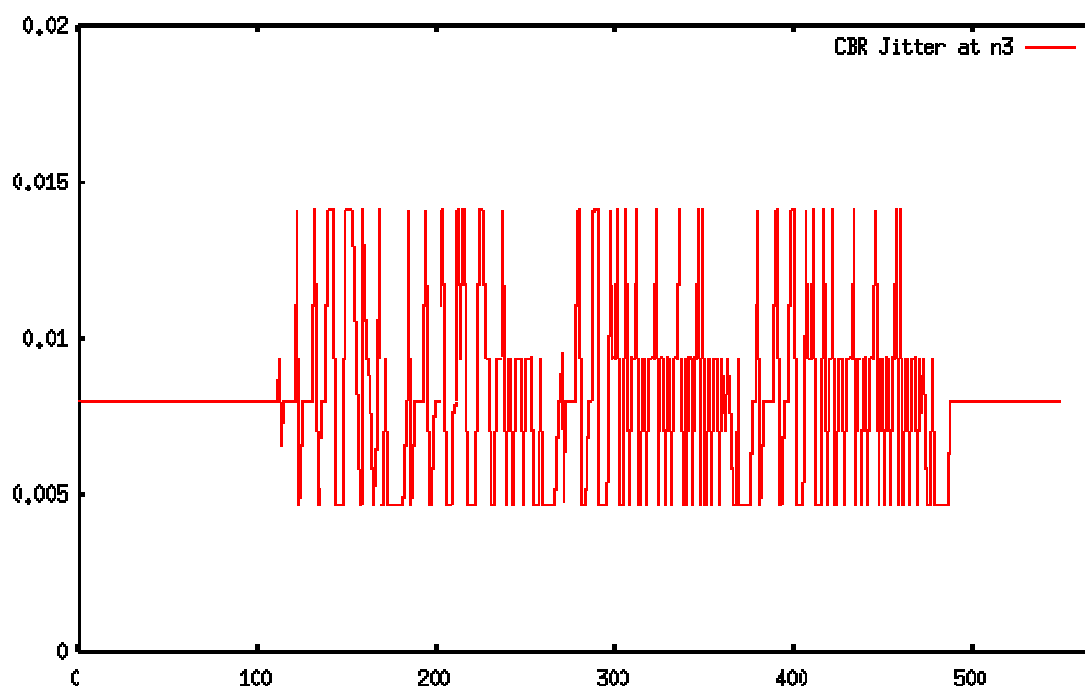
```
If (dif>0) {printf("%d\t%n" , $2, ($1 -old1) /dif);
```

```
Old1=$1; old2=$2}}>jitter.txt
```

این برنامه رخداد دریافت بسته CBR را در گره n3 انتخاب کرده، همچنین ستون ۱ که مربوط به زمان و ستون ۱۰ که مربوط به شماره سکانس است را انتخاب می کند، سپس اختلاف زمان دریافت بسته را با بسته قبلی محاسبه کرده و بر تفاضل شماره سکانس آنها تقسیم می کند. شکل زیر، گراف لرزش مربوطه که بوسیله `gnuplot` رسم شده است را نشان می دهد. محور افقی X شماره سکانس بسته و محور Y نشانگر زمان شبیه سازی بر حسب ثانیه است.

این بخش مثالی را نشان داد که چگونه می توان ردیابی را در NS انجام داد و یا چگونه آنها را تفسیر کرد، همچنین چگونه می توان اطلاعات مفید را از ردیابی بیرون کشید. در این مثال پروسه شبیه سازی بعد از عملیات شبیه سازی در خط فرمان اجرا شده است. هرچند که می توان این پردازش های نهایی را در ورودی OTCL انجام داد که این موضوع را در قسمت بعدی مطالعه خواهید کرد.


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

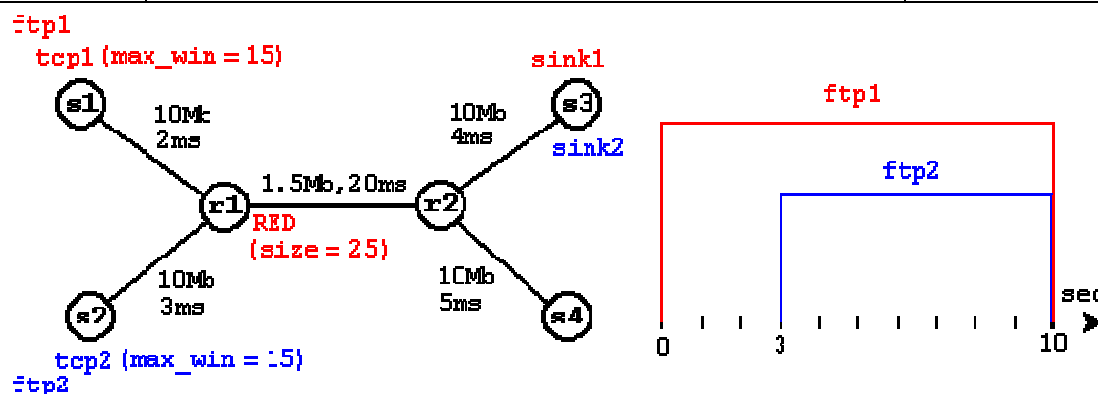


شکل ۱۲: لرزش CBR در گره n3

۹-۴-۱- مثالی از نمایشگر صف RED

این بخش مثالی از نمایش صف RED را نشان می دهد. مثال ۵ یک برنامه OTCL است که توپولوژی شبکه نشان داده شده در شکل ۱۳ را ایجاد کرده و عملیات شبیه سازی مربوط را انجام می دهد. توجه داشته باشید که برای لینک بین r1 و r2 از صف RED که توانایی داشتن ۲۵ بسته را دارد، استفاده شده است. هدف این است که ببینیم صف RED چگونه کار می کند و این عمل بوسیله اندازه گیری جریان و متوسط اندازه صف انجام می شود. برنامه زیر را در فایل به اسم red.tcl ذخیره کنید و سپس در خط فرمان عبارت ns red.tcl را اجرا کنید.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---



شکل ۱۳: تنظیمات مربوط به مثال نمایشگر صف RED

دو نکته قابل توجه در برنامه عبارتند از:


اول اینکه تابع پیشرفته create-connection برای تولید اتصالات Tcp مورد استفاده قرار گرفته است. دوم اینکه با دقت به قسمت ردیابی صف (نمایشگر) در می یابیم که این خطوط برنامه، باعث ساختن یک متغیر برای اشاره به شیء صف RED و فراخوانی توابع ردیابی برای نمایش اندازه صف جاری (curq_) و نیز اندازه متوسط صف (avg_) خواهند شد. همچنین نتایج در فایل all.q نوشته می شوند. دو فرمت خروجی ردیابی صف برای اندازه متوسط صف و اندازه صف جاری بصورت زیر می باشند.

A time ave_q_size

Q time crnt-q_size

تا اینجا تمام کارهای لازم برای نمایش صف RED به غیر از بستن فایل all.q هنگام شبیه سازی، که

جزء پروسه پردازش نهایی است، انجام شده اند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---

```

set ns [new Simulator]

set node_(s1) [$ns node]
set node_(s2) [$ns node]
set node_(r1) [$ns node]
set node_(r2) [$ns node]
set node_(s3) [$ns node]
set node_(s4) [$ns node]

$ns duplex-link $node_(s1) $node_(r1) 10Mb 2ms DropTail
$ns duplex-link $node_(s2) $node_(r1) 10Mb 3ms DropTail
$ns duplex-link $node_(r1) $node_(r2) 1.5Mb 20ms RED
$ns queue-limit $node_(r1) $node_(r2) 25
$ns queue-limit $node_(r2) $node_(r1) 25
$ns duplex-link $node_(s3) $node_(r2) 10Mb 4ms DropTail
$ns duplex-link $node_(s4) $node_(r2) 10Mb 5ms DropTail
.
.
.

set tcp1 [$ns create-connection TCP/Reno $node_(s1) TCFSink $node_(s3) 0]
$tcp1 set window_ 15
set tcp2 [$ns create-connection TCP/Reno $node_(s2) TCFSink $node_(s3) 1]
$tcp2 set window_ 15
set ftp1 [$tcp1 attach-source FTP]
set ftp2 [$tcp2 attach-source FTP]

# Tracing a queue
set redq [$ns link $node_(r1) $node_(r2)] queue
set tchan_ [open all.q w]
$redq trace curq_
$redq trace ave_
$redq attach $tchan_

$ns at 0.0 "$ftp1 start"
$ns at 3.0 "$ftp2 start"
$ns at 10 "finish"

# Define 'finish' procedure (include post-simulation processes)
proc finish {} {
    global tchan_
    set awkCode {
        {
            if ($1 == "Q" && NF>2) {
                print $2, $3 >> "temp.q";
                set end $2
            }
            else if ($1 == "a" && NF>2)
                print $2, $3 >> "temp.a";
        }
    }
    set f [open temp.queue w]
    puts $f "TitleText: red"
    puts $f "Device: Postscript"


    if { [info exists tchan_] } {
        close $tchan_
    }
    exec rm -f temp.q temp.a
    exec touch temp.a temp.q

    exec awk $awkCode all.q

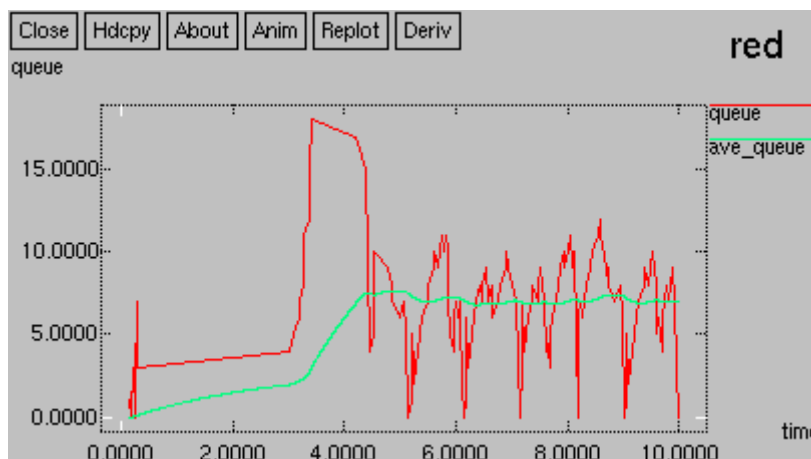
    puts $f "\"queue
    exec cat temp.q >& $f
    puts $f "\\n\"ave_queue
    exec cat temp.a >& $f
    close $f
    exec xgraph -bb -tk -x time -y queue temp.queue &
    exit 0
}

$ns run

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---


مثال ۵: برنامه ای برای ایجاد نمایشگر صف

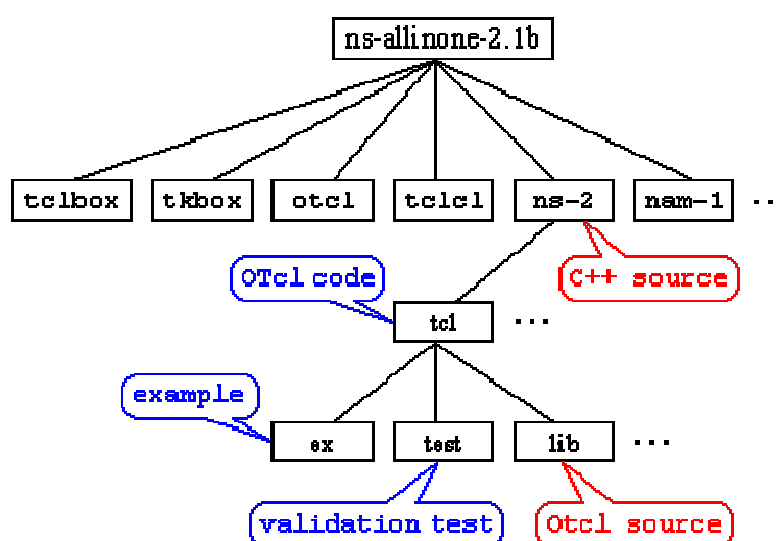


شکل ۱۴: گراف ردیابی صف Red

۱-۴-۱۰- چه چیزی را کجا پیدا کنیم؟

قبل از اینکه وارد بحث گسترش NS شویم، اجازه دهید بطور مختصر بررسی کنیم چه اطلاعاتی در کدام شاخه یا فایل ذخیره شده اند. شکل ۱۵ ساختار شاخه ای شبیه ساز NS را نشان می دهد. در اینجا فرض شده است که شما ns-allinone-2.1b را نصب کرده اید. در بین زیرشاخه های ns-allinone-2.1b، ns-2، مکان قرار گرفتن تمام اجزای شبیه ساز، چه در محیط C++ و یا چه در محیط OTCL، تست معتبر بودن برنامه های OTCL و مثالهایی از OTCL است. در این شاخه تمام برنامه های OTCL چه تست و چه مثال در داخل یک زیرشاخه از آن به نام TCL قرار گرفته است. اغلب کدهای برنامه C++ که برای زمانبندی رخداد و کلاسهای شیء اجزا شبکه هستند در شاخه اصلی ns-2 قرار دارند. بعنوان مثال اگر شما می خواهید کاربرد عوامل UDP را مشاهده کنید، باید به زیرشاخه ns-allinone-2.1b/ns-2 بروید و فایل udp.h یا udp.cc را ببینید.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---



شکل ۱۵: ساختار شاخه ها در NS

برای سلسله مراتب کلاس مربوط به اجزاء شبکه، می توانید به شکل ۶ در بخش اجزای شبکه مراجعه کنید. از حالا به بعد، فرض می شود که شما در شاخه ها ns-allinone-2.1b قرار دارید. شاخه tcl دارای چندین زیرشاخه نظیر Lib است که شامل کدهای OTCL برای اغلب قسمت های ضروری و برای اجرای NS (عامل، گره، لینک، بسته، آدرس، مسیریابی و غیره) می باشد. بقیه کدهای OTCL برای قسمت های مانند LAN ، Web و اجرای چندپختی در زیرشاخه مجزا از tcl قرار دارند. آنها در زیر لیستی از فایل های داخل شاخه ns-2/cl/lib آورده شده اند.

ns-lib.tcl

کلاس شبیه ساز و بیشتر تعاریف توابع عضو آن بجز LAN و Web و مربوط به چندپختی در این فایل قرار دارند. اگر می خواهید ببینید که کدام یک از توابع کلاس شبیه ساز در دسترس هستند و چگونه کار می کنند، باید به این قسمت مراجعه کنید.

ns-packet.tcl

در اینجا اجرای مقداردهی اولیه برای فرمت سرآمد بسته ها، انجام می شود. وقتی یک سرآمد جدید برای بسته تولید می کنید، باید این سرآمد را در این فایل ذخیره کنید تا در هنگام مقداردهی اولیه به سرآمدها، سرآمد مورد نظر شما هم در پشته سرآمدها قرار گیرد و افسست سرآمد به شما اطلاع داده شود. مثال ایجاد سرآمد جدید در قسمت (اضافه کردن کاربرد یا عامل جدید) آورده شده است.

بقیه فایل های OTCL:

سایر فایل های OTCL موجود در این شاخه، شامل اجرای اشیاء ترکیبی شبکه یا اشیاء کنترلی شبکه در ++C هستند. کاربرد FTP در OTCL قابل اجرا بوده و کد برنامه مربوط به آن در فایل ns-source.tcl آمده است. دو زیرشاخه دیگر از tcl که ممکن است برای کاربری که می خواهد طراحی انجام دهد مفید باشند، زیر شاخه های ex و test هستند. شاخه اول شامل برنامه های متعدد شبیه سازی بوده و دومی دارای برنامه هایی

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

است که نصب درست NS را بر روی کامپیوتر شما نشان خواهد داد. هنگامی که این شبیه سازیها اجرا شوند و نتایج آنها با نتایج مورد انتظار مقایسه گردد، صحت نصب صحیح NS مشخص خواهد شد.


۱-۴-۱- پیوند OTCL

گسترش NS بوسیله افزودن یک شیء شبکه پایه ای جدید، اغلب مستلزم کار پیوند OTCL با کد C++ است چرا که کلاس شیء در مسیر داده باید به دلیل کارآیی در C++ نوشته شود. این بخش پیوندهای بین C++/OTCL که در NS موجود هستند را به کمک یک مثال از تولید عامل ساده و کند (dull) که My Agent نامیده شده است و هیچ بسته ای تولید و ارسال نمی شود بیان می کند. شکل ۱۶ تا شکل ۱۹ بخشهایی از فایل C++ برای My Agent را نشان می دهند که با یکدیگر یک اجرای کامل را می سازند. در انتهای این بخش، یک برنامه OTCL به کمک آن می توانید My Agent را تست کنید، آورده شده است.

۱-۴-۲- صادر کردن کلاس از C++ به OTCL

فرض کنید که شما یک کلاس جدید شیء شبکه تولید کرده اید با فراخوانی My Agent که از کلاس Agent اخذ شده است. می توانید این امکان را فراهم آورید که نمونه ای از این کلاس را در OTCL تولید نمایید. برای انجام آن مجبورید که یک شیء پیوند تعریف کنید، بعنوان مثال بگویید My Agent Class که باید از Tcl Class اخذ شده اشد. این پیوند یک شیء OTCL با اسم خاصی (در این مثال Agent/My Agent OCL) را ایجاد می کند و بین شیء OTCL و شیء C++ (در این مثال My Agent) یک پیوند برقرار می نماید. شکل ۱۶ تعریف کلاس My Agent و تعریف کلاس پیوند را نشان می دهد.

وقتی که NS شروع بکار می کند، سازنده (constructor) را برای متغیر استاتیک class-my-agent اجرا می کند و بنابراین یک مثال از My Agent Class را تولید می کند. در این پروسه، کلاس Agent/My Agent OTCL و توابع عضو در محیط OTCL ایجاد می شوند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```
class MyAgent : public Agent {
public:
    MyAgent();
protected:
    int command(int argc, const char*const* argv);
private:
    int    my_var1;
    double my_var2;
    void    MyPrivFunc (void);
};
```

```
static class MyAgentClass : public TclClass {
public:
    MyAgentClass() : TclClass("Agent/MyAgentOtcl") {}
    TclObject* create(int, const char*const*) {
        return(new MyAgent());
    }
} class_my_agent;
```

شکل ۱۶: مثال C++ از جزء شبکه و شیء پیوند

هنگامیکه NS برای اولین بار اجرا می شود، سازنده (constructor) را برای متغیر استاتیک class-my-agent اجرا می کند و بنابراین MyAgent ایجاد می گردد. در طی این عملیات کلاس Agent/My AgentOtcl و توابع کتابخانه ای آن در محیط OTCL تولید می شوند.


۱-۴-۱۳- صادر کردن متغیر از C++ به OTCL

فرض کنید که شیء جدید C++ برنامه شما یعنی MyAgent دارای دو متغیر my_var1 و my_var2 است که شما می خواهید آنها را از طریق برنامه OTCL تغییر بدهید. برای این منظور باید برای هر کدام از متغیرهایی که می خواهید آنها را صادر کنید، یک تابع تخصیص بدهید. شکل ۱۷ چگونگی این کار را نشان می دهد. شبیه ساز NS چهار نوع مختلف از توابع را برای پنج نوع از متغیرها مورد حمایت قرار می دهد:

```
MyAgent::MyAgent() : Agent(PT_UDP) {
    bind("my_var1_otcl", &my_var1);
    bind("my_var2_otcl", &my_var2);
}
```

شکل ۱۷: مثالی از تخصیص متغیرها

- Bind() :real or integer variables
- Bind_time() :time variable
- Bind_bw() :bandwidth variable
- Bind_bool() :Boolean variable

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

۱-۴-۱۳-۱- صادر کردن فرمانهای کنترلی از C++ به OTCL

در هنگام صادر کردن متغیرها از C++ به OTCL، ممکن است بخواهید کنترل آنها را نیز به بسپارید. این کار با تعریف تابع محلی command در MyAgent که به عنوان مفسر دستورات OTCL است، انجام می شود شکل ۱۸ مثالی از تعریف تابع command برای شیئی شکل ۱۶ را نشان می دهد.

```
int MyAgent::command(int argc, const char*const* argv) {
    if(argc == 2) {
        if(strcmp(argv[1], "call-my-priv-func") == 0) {
            MyPrivFunc();
            return(TCL_OK);
        }
    }
    return(Agent::command(argc, argv));
}
```

شکل ۱۸: مثالی از مفسر دستور OTCL

۱-۴-۱۳-۲- اجرای یک دستور OTCL از C++

هنگامیکه یک شیئی جدید شبکه را در C++ ایجاد می کنید، ممکن است بخواهید یک دستور OTCL را از شیئی C++ اجرا نمایید. شکل ۱۹ نشان می دهد که تابع MyPrivFunc که یک تابع محلی از MyAgent در شکل ۱۶ است، چگونه مفسر OTCL را مجبور به چاپ مقدار متغیرهای محلی my_var1 و my_var2 می کند.


```
void MyAgent::MyPrivFunc(void) {
    Tcl& tcl = Tcl::instance();
    tcl.eval("puts \"Message From MyPrivFunc\"");
    tcl.evalf("puts \"    my_var1 = %d\"", my_var1);
    tcl.evalf("puts \"    my_var2 = %f\"", my_var2);
}
```

شکل ۱۹: اجرای دستور OTCL از شیئی C++

برای اجرای یک دستور OTCL از C++، باید به Tcl::instance() که به عنوان یک متغیر ایستا از قبل معرفی شده است، رجوع کنید. این مثال دو راه برای انتقال یک دستور OTCL به مفسر بیان می کند. برای داشتن لیست کاملی از توابعی که این انتقال را انجام می دهند، به N.S. Manual رجوع کنید.

۱-۴-۱۳-۳- اجرا، کامپایل و تست کردن MyAgent

تا این قسمت، متغیرهای ضروری پیوند OTCL در N.S. را به کمک مثال MyAgent آزمایش کردیم. از آنجایی که اجرا و آزمایش این مثال به فهم بیشتر خواننده کمک می کند، یک روش که برای کامپایل، اجرا و تست MyAgent لازم است، پیشنهاد می کنیم.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

۱- فایل ex.linkage.cc را در شاخه ns-2 ذخیره کنید.

۲- Makefile را باز کرده و فایل ex.linkage.o را به انتهای لیست اشیاء اضافه کنید.

۳- به کمک دستور make شبیه ساز N.S. را مجدداً کامپایل کنید.

۴- فایل ex.linkage.tcl که شامل دستورات OTCL برای تست MyAgent است، را به کمک دستور

ex.linkage.tcl اجرا کنید.

برنامه ex.linkage.cc و نتیجه آن را در زیر ملاحظه می کنید.

```
//Jae Chung 7-13-99
//Example of a aimple and dull Agent that
//illustrates the use of OTcl linkages
```

```
#include <stdio.h>
#include <string.h>
#include "agent.h"
```

```
class MyAgent : public Agent {
public:
    MyAgent; ()
protected:
    int command(int argc, const char*const* argv);
private:
    int    my_var1;
    double my_var2;
    void    MyPrivFunc(void);
};
```

```
static class MyAgentClass : public TclClass {
public:
    MyAgentClass() : TclClass("Agent/MyAgentOtcl") {}
    TclObject* create(int, const char*const*) {
        return(new MyAgent; ());
    }
} {class_my_agent;
```

```
MyAgent::MyAgent() : Agent(PT_UDP) {
    bind("my_var1_otcl", &my_var1);
    bind("my_var2_otcl", &my_var2);
}
```

```
int MyAgent::command(int argc, const char*const* argv) {
    if(argc == 2) {
        if(strcmp(argv[1], "call-my-priv-func") == 0) {
            MyPrivFunc();
            return(TCL_OK; (
        }
    }
    return(Agent::command(argc, argv());
}
```


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```
void MyAgent::MyPrivFunc(void ){
    Tcl& tcl = Tcl::instance();
    tcl.eval("puts \"Message From MyPrivFunc\"");
    tcl.evalf("puts \"      my_var1 = %d\\\"", my_var1);
    tcl.evalf("puts \"      my_var2 = %f\\\"", my_var2);
}
```

ex-linkage.tcl

```
# Create MyAgent (This will give two warning messages that
# no default vaules exist for my_var1_otcl and my_var2_otcl)
set myagent [new Agent/MyAgentOtcl]

# Set configurable parameters of MyAgent
$myagent set my_var1_otcl 2
$myagent set my_var2_otcl 3.14

# Give a command to MyAgent
$myagent call-my-priv-func
```

شکل ۲۰: ex.linkage.tcl

result

```
warning: no class variable Agent/MyAgentOtcl::my_var1_otcl
      see tcl-object.tcl in tclcl for info about this warning.
warning: no class variable Agent/MyAgentOtcl::my_var2_otcl

Message From MyPrivFunc
      my_var1 = 2
      my_var2 = 3.140000
```

شکل ۲۱: نتیجه مربوط به برنامه ex.linkage.tcl

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

۱-۴-۱- اضافه کردن یک کاربرد و یک عامل (Agent) جدید

۱-۴-۱-۴- هدف

می خواهیم یک کاربرد چند رسانه ای (Multimedia) بسازیم که روی یک ارتباط UDP اجرا شود و بتوانیم رفتار یک کاربرد چند رسانه ای را بصورت مجازی مشاهده و شبیه سازی کنیم.

۱-۴-۱-۲- توصیف کاربرد


در این کاربرد، فرض شده است وقتی یک اتصال برقرار می شود، فرستنده و گیرنده روی ۵ مجموعه مختلف از نحوه کد کردن و ارسال، که آنها را با اعداد ۰ تا ۴ نشان می دهیم، اتفاق نظر دارند. برای راحتی فرض کنیم که هر یک از فرستنده یا گیرنده ها دارای نرخ دریافت و ارسال ثابتی هستند و طول بسته ها نیز یکسان است و به نحوه کد کردن ربطی ندارند.

این کاربرد چند رسانه ای به این صورت کار می کند که در ابتدا فرستنده با نرخ ارسال صفر شروع می کند و سپس نرخ ارسال خودش را با مقداری که گیرنده به آن اطلاع می دهد، گیرنده مسئول تراکم شبکه و تعیین نرخ ارسال مناسب می باشد. برای نمایش تراکم شبکه، از یک بسته ساده که در هر RTT از آن استفاده می شود، کمک گرفته می شود. اگر تراکم در شبکه مشاهده شود آنگاه گیرنده مقدار پارامتر Scale را به نصف کاهش می دهد و این مقدار کاهش یافته را به اطلاع فرستنده نیز می رساند. اما اگر بسته ساده گم نشود آنگاه پارامتر Scale را یکی افزایش داده و این مقدار را به فرستنده نیز اطلاع می دهد.

۱-۴-۱-۳- تحلیل مشکلات

قبل از پیاده سازی این کاربرد، کاربرد مربوط به عامل UDP آزمایش شده و یک مشکل اساسی مشاهده شد. از آنجائی که عامل UDP وظیفه ارسال بسته های شبکه را برعهده دارد، تمام اطلاعات لازم برای انتقال، باید به صورت رشته ای از داده، در اختیار عامل UDP قرار بگیرد.

UDP فقط بسته هایی را که دارای پشته سرآمد هستند ملاحظه می کند، بنابراین باید پیاده سازی UDP را تغییر دهیم تا قادر باشیم که داده های دریافت شده از کاربردها را نیز اضافه کنیم. از آنجائیکه ما بعداً می خواهیم از این کاربرد در مکانیزم مدیریت صف مسیریاب IP نیز استفاده کنیم، بنابراین بدنبال راهی هستیم تا این رشته چندپخشی را از سایر رشته ها متمایز کنیم. برای این منظور باید عوامل UDP طوری تغییر کند که بتواند داده ها را در یکی از سرآمدهای IP ای که مورد استفاده قرار نگرفته است، ذخیره کند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۴-۱- طراحی و پیاده سازی


برای این کاربرد از CBR و اعمال تغییراتی روی آن برای داشتن پنج سطح Scale (0-4) استفاده شده است. نام این کلاس از کاربرد را MmApp می گذاریم که یک کلاس فرزند از کلاس Application است. در شکل های زیر مراحل طراحی و پیاده سازی نمایش داده شده است.

```
// Multimedia Header Structure
struct hdr_mm {
    int ack;        // is it ack packet?
    int seq;        // mm sequence number
    int nbytes;     // bytes for mm pkt
    double time;    // current time
    int scale;      // scale (0-4) associated with data rates

    // Packet header access functions
    static int offset_;
    inline static int& offset() { return offset_; }
    inline static hdr_mm* access(const Packet* p) {
        return (hdr_mm*) p->access(offset_);
    }
};
```

```
// Multimedia Header Class
static class MultimediaHeaderClass : public PacketHeaderClass {
public:
    MultimediaHeaderClass() : PacketHeaderClass("PacketHeader/Multimedia",
                                                sizeof(hdr_mm)) {
        bind_offset(&hdr_mm::offset_);
    }
} class_mmhdr;
```

شکل ۲۲: ساختار سرآمد MM و کلاس آن

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---


```
enum packet_t {
    PT_TCP,
    ...
    PT_Multimedia,
    PT_NTTYPE // This MUST be the LAST one
};

class p_info {
public:
    p_info() {
        name_[PT_TCP] = "tcp";
        ...
        name_[PT_Multimedia] = "Multimedia";
        name_[PT_NTTYPE] = "undefined";
    }
    ...
};
```

شکل ۲۳: اضافه کردن به فایل packet که از نوع C++ است.

```
foreach prot {
    AODV
    ...
    Multimedia
} {
    add-packet-header $prot
}
```

شکل ۲۴: اضافه کردن به فایل ns-packet.tcl که از نوع OTCL است.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```

class SendTimer : public TimerHandler {
public:
    SendTimer(MmApp* t) : TimerHandler(), t_(t) {}
    inline virtual void expire(Event*);
protected:
    MmApp* t_;
};

void SendTimer::expire(Event*)
{
    t_->send_mm_pkt();
}

```

```

class MmApp : public Application {
public:
    MmApp();
    ...
private:
    ...
    SendTimer snd_timer_;
    ...
};

```

```

MmApp::MmApp() : running_(0), snd_timer_(this), ack_timer_(this)
{
    bind_bw("rate0_", &rate[0]);
    ...
    bind_bw("rate4_", &rate[4]);
    bind("pktsize_", &pktsize_);
    bind_bool("random_", &random_);
}

```

```


void MmApp::send_mm_pkt()
{
    hdr_mm mh_buf;

    if (running_) {
        ...
        agent_->sendmsg(pktsize_, (char*) &mh_buf); // send to UDP

        // Reschedule the send_pkt timer
        double next_time_ = next_snd_time();
        if(next_time_ > 0) snd_timer_.resched(next_time_);
    }
}

```

شکل ۲۵: ارسال اجرای تایمر

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
---	---	---

```

class Agent : public Connector {
public:
    Agent(int pktType);
    ...
    virtual int supportMM() { return 0; }
    virtual void enableMM() {}
    virtual void sendmsg(int nbytes, const char *flags = 0);
    virtual void send(int nbytes) { sendmsg(nbytes); }
    ...
}

```

شکل ۲۶: افزودن دو تابع محلی به کلاس Agent

```

class Application : public Process {
public:
    Application();
    virtual void send(int nbytes);
    virtual void recv(int nbytes);
    virtual void recv_msg(int nbytes, const char *msg = 0)();
    virtual void resume();
    ...
};

```

شکل ۲۷: افزودن دو تابع محلی به کلاس Application


```

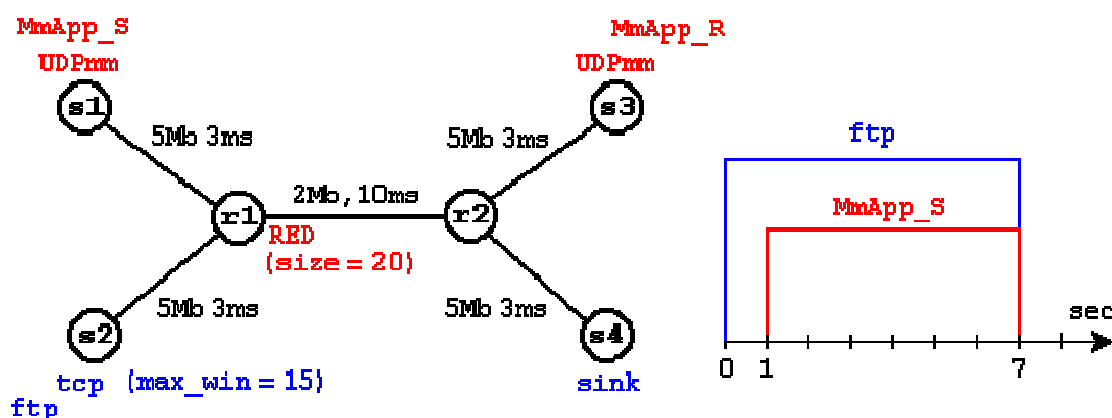
...
Application/MmApp set rate0_ 0.3mb
Application/MmApp set rate1_ 0.6mb
Application/MmApp set rate2_ 0.9mb
Application/MmApp set rate3_ 1.2mb
Application/MmApp set rate4_ 1.5mb

Application/MmApp set pktsize_ 1000
Application/MmApp set random_ false
...

```

شکل ۲۸: تنظیم مقادیر اولیه پارامترها

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---



شکل ۲۹: توپولوژی و سناریوی شبیه سازی برنامه Test


۱-۴-۱- اضافه کردن یک صف جدید

۱-۴-۱-۱- هدف

اضافه کردن یک صف جدید برای ساخت صف خروجی مربوط به مسیریاب (router) drop-tail از خاصیت زمان بندی round-robin dequeue برای حق تقدم ۱۵ بسته موجود در یک صف استفاده می کنند. این وضعیت هنگامی اتفاق می افتد که ۱۵ بسته که دارای حق تقدم بیشتری هستند به همراه دیگر بسته ها در یک صف قرار دارد، با این کار قدیمی ترین بسته از هر نوع با اولویت، به ترتیب از صف خارج می شوند.

۱-۴-۱-۲- طراحی

صف مورد نظر دارای دو صف از نوع FIFO است که آنها را LQ1 و LQ2 می نامند. اندازه کلی این صفها مساوی اندازه صف فیزیکی (PQ) است. بعبارت دیگر $LQ1 + LQ2 = PQ$ می باشد. برای ایجاد رفتار متناسب با صف drop-tail، وقتی یک بسته وارد صف می شود (enqueue)، مدیریت صف چک می کند که آیا اندازه $LQ1 + LQ2$ از حداکثر مجاز که PQ است، کمتر می باشد یا خیر. اگر این اندازه کمتر بود، این بسته وارد صف مناسب خواهد شد. برای انجام عملیات خارج کردن بسته ها از صف (dequeue) بر اساس زمان بندی round-robin، مدیریت صف سعی می کند یک بسته را از یکی از صفهای LQ1 یا LQ2 خارج کند و بسته دیگر را از صف دیگر خارج نماید. بعبارت دیگر بسته های موجود در صفهای LQ1 و LQ2 با نرخ یک به یک خارج می شوند و این عمل تا هنگامی که هر دو صف دارای بسته باشند ادامه پیدا خواهد کرد.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۴-۱۵-۳- پیاده سازی

این شیء صف DtRrQueue (Drop-Tail Round-Robin Queue) با یک اسم C++ نامگذاری می شود که از کلاس Queue اخذ شده است. نام OTCL معادل آن Queue/DTRR است. هنگامی که تابع recv که در کلاس Queue اجرا شده است، یک بسته دریافت می کند، تابع enqueue را صدا می زند و اگر شیء اتصال مسدود نشده باشد تابع dequeue را نیز صدا می کند. بنابراین مجبور هستیم توابع enqueue و dequeue را برای کلاس DtRrQueue بنویسیم. شکل ۲۹ تعریف کلاس DtRrQueue و توابع آنرا نشان می دهد.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

class DtRrQueue : public Queue {
public:
    DtRrQueue() {
        q1_ = new PacketQueue;
        q2_ = new PacketQueue;
        pq_ = q1_;
        deq_turn_ = 1;
    }

protected:
    void enqueue(Packet*);
    Packet* deque();

    PacketQueue *q1_;    // First FIFO queue
    PacketQueue *q2_;    // Second FIFO queue
    int deq_turn_;       // 1 for First queue 2 for Second
};

```

```

void DtRrQueue::enqueue(Packet* p)
{
    hdr_ip* iph = hdr_ip::access(p);

    // if IPv6 priority = 15 enqueue to queue1
    if (iph->prio_ == 15) {
        q1_->enqueue(p);
        if ((q1_->length() + q2_->length()) > qlim_) {
            q1_->remove(p);
            drop(p);
        }
    }
    else {
        q2_->enqueue(p);
        if ((q1_->length() + q2_->length()) > qlim_) {
            q2_->remove(p);
            drop(p);
        }
    }
}

```

```

Packet* DtRrQueue::deque()
{
    Packet *p;

    if (deq_turn_ == 1) {
        p = q1_->deque();
        if (p == 0) {
            p = q2_->deque();
            deq_turn_ = 1;
        }
        else
            deq_turn_ = 2;
    }
    else {
        p = q2_->deque();
        if (p == 0) {
            p = q1_->deque();
            deq_turn_ = 2;
        }
        else
            deq_turn_ = 1;
    }

    return (p);
}

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

شکل ۳۰: پیاده سازی کلاس DtRr Queue

۱-۴-۱۵-۴- آزمایش

در این قسمت از برنامه شبیه سازی برای آزمایش MmApp روی UDPmm که در قسمت قبل آمده است استفاده می کنیم، فقط صف RED را به صف DRTT تغییر می دهیم که این صف مربوط به لینک بین r1-r2 است. تغییرات اعمال شده در شکل ۳۱ آمده است.

```
Set ns [new Simulator]
...
$ns duplex-link $node_(s1) $node_(r1) 5Mb 3ms DropTail
$ns duplex-link $node_(s2) $node_(r1) 5Mb 3ms DropTail
$ns duplex-link $node_(r1) $node_(r2) 2Mb 10ms DTRR
$ns duplex-link $node_(s3) $node_(r2) 5Mb 3ms DropTail
$ns duplex-link $node_(s4) $node_(r2) 5Mb 3ms DropTail

#Set DTRR queue size to 20
$ns queue-limit $node_(r1) $node_(r2) 20
...

#Simulation Scenario
$ns at 0.0 "$ftp start"
$ns at 1.0 "$mmapp_s start"
$ns at 7.0 "finish"

$ns run
```

شکل ۳۱: برنامه آزمایش Dr Rr Queue


۱-۴-۱۶- مثال LAN

این بخش شامل مثالی از برنامه شبیه سازی مربوط به LAN است. توپولوژی این شبکه و مراحل شبیه سازی آن به نمایش درآمده است. برنامه زیر را اجرا کنید و نتایج آنرا ملاحظه کنید.

```
#Author: Jae Chung
#Date: 7/20/99
#
#This file is modified from
" #ns-2/tcl/ex/lantest.tcl"

set opt(tr) "out.tr"
set opt(namtr) "out.nam"
set opt(seed) .
set opt(stop) 5
set opt(node) 8

set opt(qsize) 100
set opt(bw) 10Mb
set opt(delay) 1ms
set opt(ll) LL
set opt(ifq) Queue/DropTail
```

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```

set opt(mac)    Mac/Csma/Ca
set opt(chan)   Channel
set opt(tcp)    TCP/Reno
set opt(sink)   TCPSink

set opt(app)    FTP

proc finish {} {
    global ns opt trfd ntrfd

    $ns flush-trace
    close $trfd
    close $ntrfd
    exec nam $opt(namtr) &
    exit 0
}

proc create-trace {} {
    global ns opt

    set trfd [open $opt(tr) w]
    $ns trace-all $trfd
    return $trfd
}

proc create-namtrace {} {
    global ns opt

    set ntrfd [open $opt(namtr) w]
    $ns namtrace-all $ntrfd
}

proc create-topology {} {
    global ns opt
    global lan node source node0

    set num $opt(node)
    for {set i 0} {$i < $num} {incr i} {
        set node($i) [$ns node[
            lappend nodelist $node($i)
        ]
    }

    set lan [$ns newLan $nodelist $opt(bw) $opt(delay)\
        -llType $opt(ll) -ifqType $opt(ifq)\
        -macType $opt(mac) -chanType $opt(chan)]

    set node0 [$ns node]
    $ns duplex-link $node0 $node(0) 2Mb 2ms DropTail

    $ns duplex-link-op $node0 $node(0) orient right
}

##MAIN##

set ns [new Simulator]
set trfd [create-trace]

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

```
set ntrfd [create-namtrace]
```

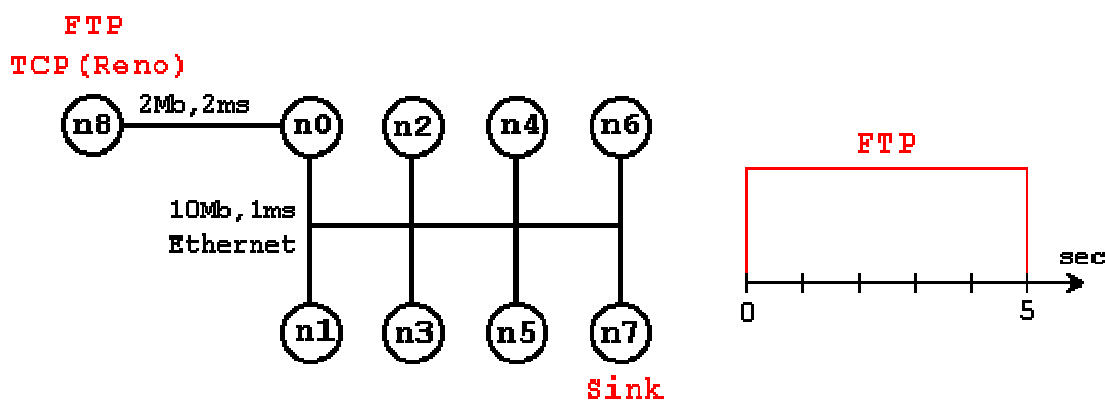
```
create-topology
```

```
set tcp0 [$ns create-connection TCP/Reno $node0 TCPSink $node(7) 0]
$tcp0 set window_ 15
```

```
set ftp0 [$tcp0 attach-app FTP]
```

```
$ns at 0.0 "$ftp0 start"
$ns at $opt(stop) "finish"
```

```
$ns run
```



شکل ۳۲: توپولوژی شبکه LAN و شبیه سازی آن

۱-۴-۱۷- مثال چندپخششی (Multicasting)

این قسمت شامل مثالی از شبیه سازی عملیات ارسال بصورت چندپخششی می باشد که متن برنامه و نتایج NAM آن را مشاهده خواهید کرد. برنامه زیر را با اسم ex.mcast.tcl ذخیره و اجرا کنید. این برنامه از پنجمین خودآموز VINT/NS آورده شده است.


```
#Polly Huang 8-7-98

set ns [new Simulator]
$ns multicast

set f [open out.tr w]
$ns trace-all $f
$ns namtrace-all [open out.nam w]

$ns color 1 red
#prune/graft packets
$ns color 30 purple
$ns color 31 green

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

```
set n3 [$ns node]
```

```
#Use automatic layout
$ns duplex-link $n0 $n1 1.5Mb 10ms DropTail
$ns duplex-link $n1 $n2 1.5Mb 10ms DropTail
$ns duplex-link $n1 $n3 1.5Mb 10ms DropTail
```

```
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n1 $n3 orient right-down
$ns duplex-link-op $n0 $n1 queuePos 0.5
```

```
set mrthandle [$ns mrtproto DM{} ]
```

```
set cbr0 [new Application/Traffic/CBR]
set udp0 [new Agent/UDP]
$cbr0 attach-agent $udp0
$ns attach-agent $n1 $udp0
$udp0 set dst_ 0x8001
```

```
set cbr1 [new Application/Traffic/CBR]
set udp1 [new Agent/UDP]
$cbr1 attach-agent $udp1
$udp1 set dst_ 0x8002
$udp1 set class_ 1
$ns attach-agent $n3 $udp1
```

```
set rcvr [new Agent/LossMonitor]
$#ns attach-agent $n3 $rcvr
$ns at 1.2 "$n2 join-group $rcvr 0x8002"
$ns at 1.25 "$n2 leave-group $rcvr 0x8002"
$ns at 1.3 "$n2 join-group $rcvr 0x8002"
$ns at 1.35 "$n2 join-group $rcvr 0x8001"
```


```
$ns at 1.0 "$cbr0 start"
$ns at 1.1 "$cbr1 start"
```

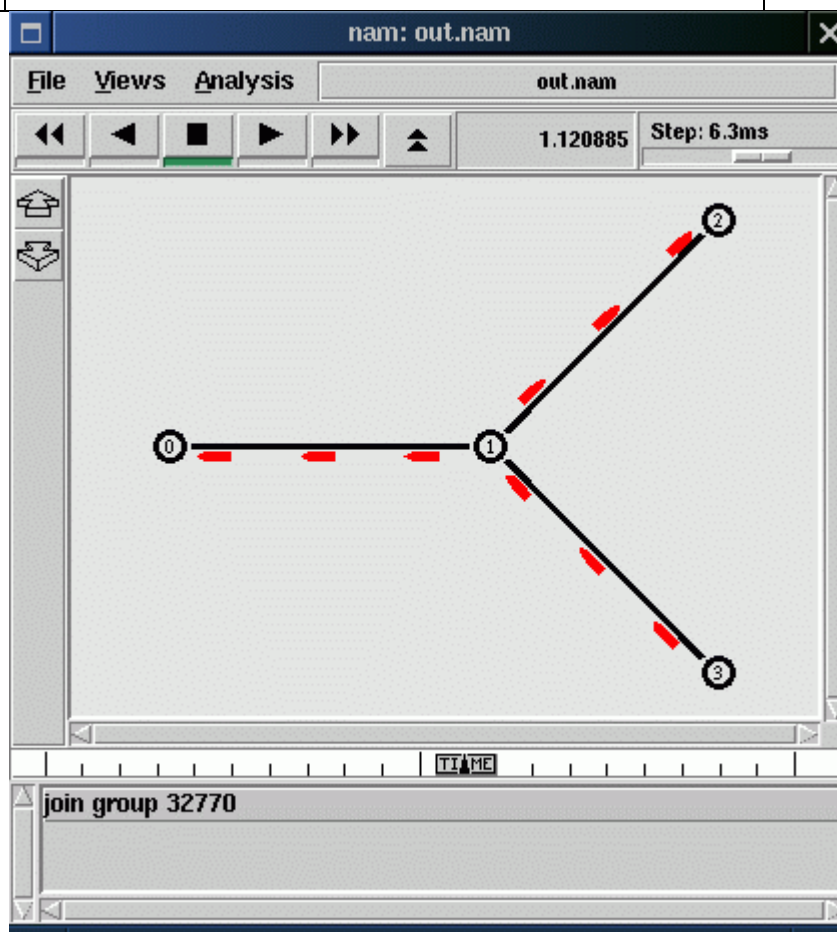
```
$ns at 2.0 "finish"
```

```
proc finish {} {
    global ns
    $ns flush-trace

    puts "running nam"...
    exec nam out.nam&
    exit 0
}
```

```
$ns run
```

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
---	---	---



شکل ۳۳: صفحه NAM مربوط به شبیه سازی ارسال چندپخشی


۱-۴-۱۸- مثال وب سرور

برنامه مربوط به شبیه سازی web server، توپولوژی و نتایج مربوط به آن در این قسمت آورده شده است. برنامه های زیر را با اسم های ex.web.tcl و dumbbell.tcl ذخیره و اجرا کنید و نتایج آنها را ملاحظه کنید. این مثالها هم از همچنین خودآموز NS آورده شده اند. توجه کنید که باید در ابتدای برنامه مسیر مربوط به http-mod.tcl را تغییر دهید تا این برنامه روی کامپیوتر شما اجرا شود.

```
#Polly Huang 8-7-98

#Initial setup
source ~/ns-allinone-2.1b4a/ns-2/tcl/http/http-mod.tcl
source dumbbell.tcl
global num_node n

set ns [new Simulator]
$ns set-address 7 24 ;# set-address <bits for node address> <bits for port>
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

```
#set up colors for nam
for {set i 1} {$i <= 30} {incr i} {
    set color [expr $i % 6]
    if {$color == 0} {
        $ns color $i blue
    } elseif {$color == 1} {
        $ns color $i red
    } elseif {$color == 2} {
        $ns color $i green
    } elseif {$color == 3} {
        $ns color $i yellow
    } elseif {$color == 4} {
        $ns color $i brown
    } elseif {$color == 5} {
        $ns color $i black
    }
}

#Create nam trace and generic packet trace
$ns namtrace-all [open out.nam w]
#trace-all is the generic trace we've been using
$ns trace-all [open out.tr w]

create_topology

#####Modify From Here#####
##Number of Pages per Session
set numPage 10
set httpSession1 [new HttpSession $ns $numPage [$ns picksrc[]]
set httpSession2 [new HttpSession $ns $numPage [$ns picksrc[]


##Inter-Page Interval
##Number of Objects per Page
##Inter-Object Interval
##Number of Packets per Object
##have to set page specific attributes before createPage
##have to set object specific attributes after createPage
$httpSession1 setDistribution interPage_ Exponential 1 ;#in sec
$httpSession1 setDistribution pageSize_ Constant 1 ;# number of
objects/page
$httpSession1 createPage
$httpSession1 setDistribution interObject_ Exponential 0.01 ;# in sec
$httpSession1 setDistribution objectSize_ ParetoII 10 1.2 ;# number of
packets

#uses default
$httpSession2 createPage

$ns at 0.1 "$httpSession1 start" ;# in sec as well
$ns at 0.2 "$httpSession2 start"

$ns at 30.0 "finish"

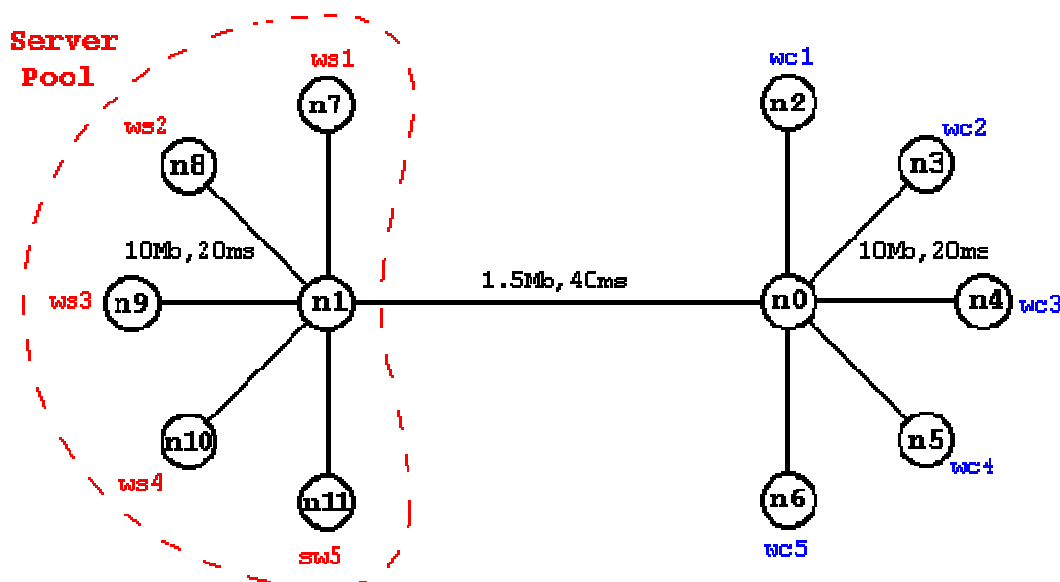
proc finish {} {
    global ns
    $ns flush-trace
    puts "running nam"...
    #    exec to run unix command
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```
exec nam out.nam&
exit 0
```

```
}
```

```
#Start the simualtion
$ns run
```



شکل ۳۴: توپولوژی شبکه web server و شبیه سازی آن


```
#Created by Polly Huang
#Simple 4-node star topology
#      8   7       2   3
#      \  |       |  /
#      9 --1 ----- 0 - 4
#      /  |       |  \
#      10  11      6   5

proc create_topology {} {
    global ns n num_node
    set num_node 12

    for {set i 0} {$i < $num_node} {incr i} {
        set n($i) [$ns node]
    }

    $ns set src_ [list 2 3 4 5 6]
    $ns set dst_ [list 7 8 9 10 11]

    #EDGES (from-node to-node length a b):
    $ns duplex-link $n(0) $n(1) 1.5Mb 40ms DropTail
    $ns duplex-link $n(0) $n(2) 10Mb 20ms DropTail
    $ns duplex-link $n(0) $n(3) 10Mb 20ms DropTail
    $ns duplex-link $n(0) $n(4) 10Mb 20ms DropTail
```


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

$ns duplex-link $n(0) $n(5) 10Mb 20ms DropTail
$ns duplex-link $n(0) $n(6) 10Mb 20ms DropTail
$ns duplex-link $n(1) $n(7) 10Mb 20ms DropTail
$ns duplex-link $n(1) $n(8) 10Mb 20ms DropTail
$ns duplex-link $n(1) $n(9) 10Mb 20ms DropTail
$ns duplex-link $n(1) $n(10) 10Mb 20ms DropTail
$ns duplex-link $n(1) $n(11) 10Mb 20ms DropTail


$ns duplex-link-op $n(0) $n(1) orient left
$ns duplex-link-op $n(0) $n(2) orient up
$ns duplex-link-op $n(0) $n(3) orient right-up
$ns duplex-link-op $n(0) $n(4) orient right
$ns duplex-link-op $n(0) $n(5) orient right-down
$ns duplex-link-op $n(0) $n(6) orient down
$ns duplex-link-op $n(1) $n(7) orient up
$ns duplex-link-op $n(1) $n(8) orient left-up
$ns duplex-link-op $n(1) $n(9) orient left
$ns duplex-link-op $n(1) $n(10) orient left-down
$ns duplex-link-op $n(1) $n(11) orient down
}
#end of create_topology

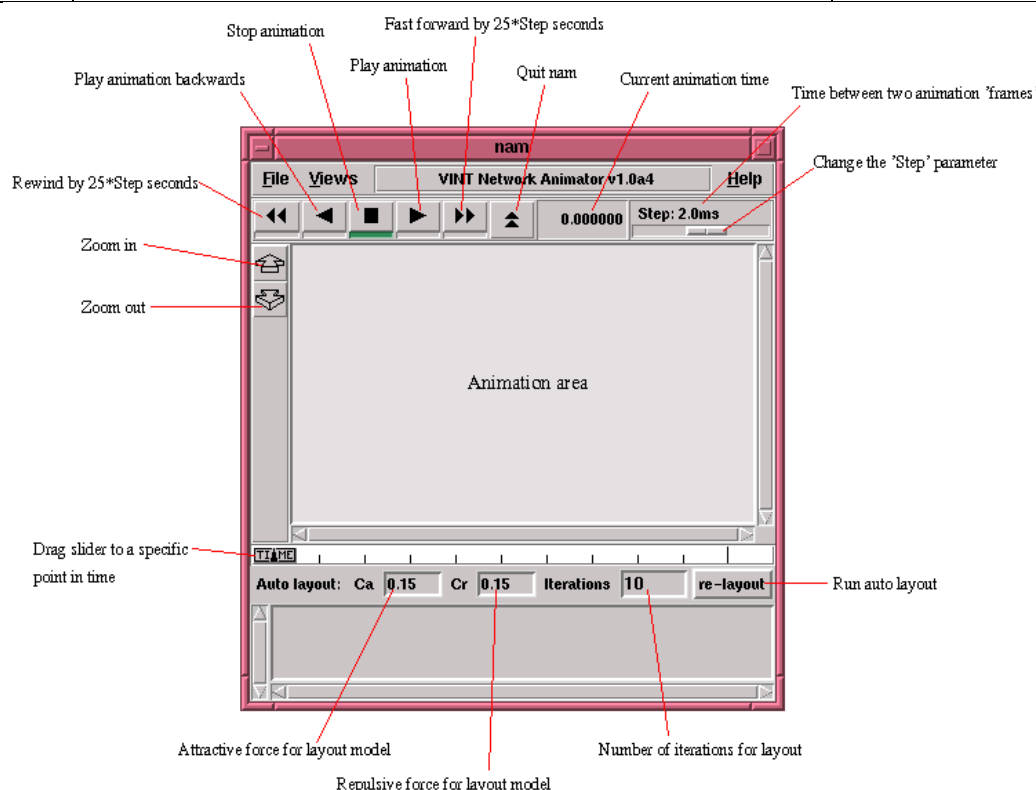
```

۱-۵- معرفی برنامه NAM

Nam یک TCL/TK بر مبنای ابزار animation برای دیدن ترسیم شبیه سازی و دنیای واقعی پیگیری بسته های داده می باشد. قدم اول در استفاده از nam، ایجاد فایل trace می باشد. این فایل باید شامل اطلاعات توپولوژی گره ها، لینک ها و packet trace باشد. غالباً فایل trace بوسیله ns تولید می شود. در طول شبیه سازی ns کاربران می توانند شکل توپولوژی، اطلاعات طرح بندی و packet trace را با استفاده از وقایع فایل trace در ns تولید کنند. هنگامیکه فایل trace تولید می شود، آماده نمایش از طریق nam می باشد. به محض شروع trace، برنامه nam فایل را می خواند و توپولوژی را ایجاد می کند. سپس پنجره را pop up می کند و اگر لازم باشد طرح ریزی می کند و تا زمان رسیدن اولین بسته در فایل trace صبر می کند

Nam یک نمایش دهنده نموداری توپولوژی شبکه و جریان داده است. با استفاده از این ابزار می توان مسیر تک تک بسته ها و حتی نوع و طول آنها را در حین عملکرد سیستم مشاهده کرد.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---




شکل ۳۵: معرفی محیط ابزار Network Animator

شکل ۳۵ محیط ابزار Nam را نشان می دهد. با این نرم افزار می توان علاوه بر نمایش نتیجه شبیه سازی، توپولوژی های ساده را نیز تولید نمود. ابزارهای بهتری نیز نظیر Nscript برای تولید توپولوژی و ترافیک شبکه وجود دارند که از GUI^۱ مناسبی برخوردار هستند و تا حد زیادی کاربر را از نوشتن کدهای Tcl بی نیاز می کنند. با این وجود کد Tcl کارآمدترین روش پیاده سازی شبکه های پیچیده بوده و امکانات بیشتری را در اختیار کاربر قرار می دهد. شکل زیر یک نمونه کد Script و تصویری از نمایش در حال جریان NAM از این ارسال داده ساده را نمایش می دهد.

در NS امکان تعریف پهنای باند، تاخیر، نوع پروتکل لایه Data link (مانند ARQ)، نوع مدیریت صف برای هر خط و نوع آدرس دهی (MAC) و نحوه ارسال داده روی خط مشترک (CSMA/CD) نیز وجود دارد. امکان اعمال رویداد در حین شبیه سازی و پویا نمودن شبیه سازی نیز وجود دارد. برای مثال می توان یک اتصال را در لحظه ای مشخص قطع نمود و یا یک خط lossy تعریف نمود.

LAN و Wireless LAN در NS به عنوان یک نود در نظر گرفته می شوند، چرا که در توپولوژی LAN تمامی رایانه ها به یک خط واحد متصل هستند که می توان آنرا یک نود میانی تصور نمود. بنابراین نوع

^۱ Graphical User Interface

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

دیگری از نود بنام LAN node در NS تعریف شده است که امکان شبیه سازی شبکه های محلی را فراهم می کند.

مسیریابی (Routing) بطور پیش فرض توسط NS انجام می شود. NS برای آدرس دهی و راهیابی در شبکه از یک سیستم تعیین آدرس خاص استفاده می کند. آدرس در NS بطور پیش فرض یک رشته ۶۴ بیتی است که ۳۲ بیت با ارزش کمتر^۱ آن معرف port-id و از ۳۲ بیت نیمه بالاتر، بیت با ارزش بالاتر برای Multicasting و ۳۱ بیت بقیه برای مشخص نمودن node-id بکار می رود. کاربر معمولاً با این آدرس دهی درونی سر و کار ندارد چراکه نام نود ها، معرف آدرس آنهاست.

^۱ Least Significant Bits



عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2

دانشگاه علم و صنعت ایران
مرکز آموزش های الکترونیکی



loaded duplex connection Tcl Code -

```
#Create a simulator object
set ns [new Simulator]

#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
    global ns nf
    $ns flush-trace
    close $nf
    exec nam out.nam &
    exit 0
}

#Create two nodes
set n0 [$ns node]
set n1 [$ns node]

#Create a duplex link between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail

#Create a UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

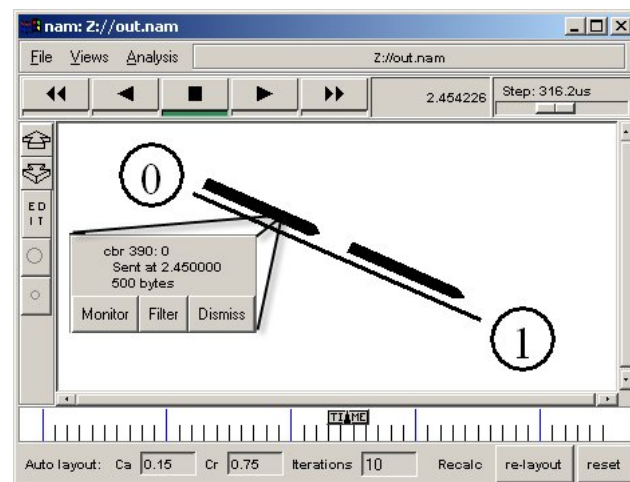
# Create a CBR traffic and attach it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize 500
$cbr0 set interval 0.005
$cbr0 attach-agent $udp0

#Create a traffic sink and attach it to node n1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

#Connect the traffic source to the traffic sink
$ns connect $udp0 $null0

#Schedule events for the CBR agent
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 5 seconds of
#simulation time
$ns at 5.0 "finish"

#Run the simulation
$ns run
```



شکل ۳۶: خروجی Tcl Code در Nam . بسته ها بصورت پیکانهایی نمایش داده می-

شوند

xGRAPH وسیله رسم نمودار در NS است. این ابزار همانند Nam بر روی فایل خروجی حاصل از شبیه سازی عمل می کند. Xgraph با استفاده از فایل trace تولید شده توسط NS قادر است مقادیر عددی را به شکل نموداری نشان دهد. این ابزار تنها برای نسخه Linux، تهیه شده و کاربران Windows باید از ابزار وابسته به MATLAB ای بنام Gnuplot و یا Tracegraph استفاده نمایند.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۱-۶- دستورات اولیه در NS2

دستوراتی که در این بخش می آید باید توسط ویرایشگر pseditor در یک فایل وارد شود و در انتها با پسوند tcl ذخیره شود. برای شبیه سازی، در ابتدا باید یک نمونه از شی simulator ایجاد گردد. این کار با دستور زیر انجام می گیرد.

```
Set ns [new simulator]
```

برای ذخیره اطلاعات مورد استفاده از namtrace باید فایلی باز شود. برای این کار از دستور زیر استفاده می شود.

```
Set nf [open out.nam w]
```

این دستور فایل out.nam را برای نوشتن باز میکند. و nf ، handler آن می باشد. یعنی برای اشاره به فایل خروجی از nf استفاده خواهد شد.

برای اینکه تمام خروجی ای مرتبط با nam به فایل nf منتقل شود از دستور زیر استفاده می شود.

```
$ns namtrace-all $nf
```

در خاتمه شبیه سازی از تابع finish استفاده می شود که فایلهای شبیه سازی را بسته، بافرها را آزاد کرده و سایر کارهای لازم را انجام دهد.

```
proc finish {} {
global nf ns
$ns flush-trace
close $nf
exec nam out.nam
exit 0
}
```

در این تابع، پس از خاتمه شبیه سازی و بستن فایل out.nam برنامه nam با ورودی out.nam صدا زده می شود. برای خاتمه شبیه سازی باید تابه finish صدا شود.

```
$ns at 5.0 "finish"
```

همانطور که دیده می شود برای زمانبندی رویدادها از کلمه کلیدی at می توان استفاده کرد.

در انتها برای شروع شبیه سازی از دستور زیر استفاده می شود.

```
$ns run
```


۱-۶-۱- مدل کردن یک توپولوژی ساده: دو گره با یک لینک

برای شروع از یک توپولوژی ساده که در آن دو گره بایک لینک به هم متصل شده اند استفاده می شود.

کد این دستورات باید قبل از \$ns run یا حتی بهتر است قبل از \$ns at 5.0 "finsh" اضافه شوند.

در ابتدا باید گره ها تعریف کرد:

```
set n0 [$ns node]
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

set n1 [\$ns node]

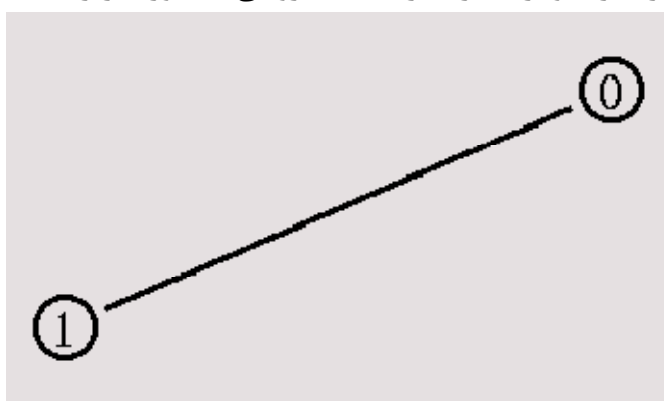
این دو دستور، گره را ایجاد کرده و آنها را به n0 و n1 نسبت می دهند.

برای تعریف لینک بین دو گره از دستور زیر استفاده می شود:

\$ns duplex-link \$n0 \$n1 1mb 10ms DropTail

این دستور یک لینک بین دو گره n0 و n1 ایجاد می کند که از نوع دو طرفه می باشد. پهنای باند 1 mbit/s و تاخیر 10ms داشته و دارای صفی از نوع DropTail است. در صف DropTail در صورت ارسال ترافیک بیش از ظرفیت خط، بسته ها از انتهای صف حذف خواهند شد. با انواع دیگر صف بعداً آشنا خواهیم شد.

حال این دستورات ذخیره کرده و برنامه را اجرا کنید. خروجی به صورت زیر مشاهده می شود:



در این بخش می خواهیم ترافیکی را از n0 به n1 ارسال کنیم.

در ns برا ی ارسال و دریافت داده ها از عاملها استفاده می شود. برای منبع ترافیک ۱ و مقصد ترافیک ۲

عامل های جداگانه ایی تعریف می شود.

#Create a UDP agent and attach it to node n0

set udp0 [new Agent/UDP]

\$ns attach-agent \$n0 \$udp0

#Create a CBR traffic source and attach it to udp0

set cbr0 [new Application/Traffic/CBR]

\$cbr0 set packet_size_500

\$cbr0 set interval_0.005

\$cbr0 attach-agent \$udp0

سه خط اول سبب ایجاد یک UDP agent و نسبت دادن آن به n0 می شود. خطوط بعد یک منبع


ترافیک CBR ایجاد کرده و به udp0 نسبت می دهد. اندازه بسته ها ۵۰۰ بایت بوده و هر ۰.۰۰۵ ثانیه یک بسته ارسال می شود.

دو دستور بعد یک عامل null که به عنوان گیرنده ترافیک عمل می کند ایجاد کرده و آن را به n1 نسبت

می دهند.

^۱ Traffic Source

^۲ Traffic Sink

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

حال دو عامل به هم متصل می شوند تا ترافیک ارسالی توسط n0 در n1 دریافت شود:

```
$ns connect $udp0 $null0
```

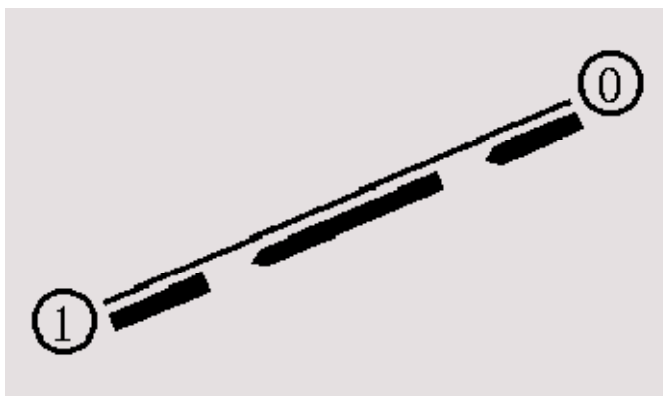
شاید این سؤال مطرح شود که ترافیک چه زمانی شروع به ارسال شده و چه زمانی متوقف شود؟ پاسخ این سؤال در زیر داده شده است.

برای کنترل منبع ترافیک از دستورات زیر استفاده می کنیم. قبل از (\$ns at 5.0 finish)

```
$ns at 0.5 "$cbr0 start"
```

```
$ns at 4.5 "$cbr0 stop"
```

حال می توانیم شبیه سازی را انجام دهیم . وقتی در پنجره باز شده توسط nam بر روی دکمه play کلیک کنیم مشاهده می شود در ثانیه ۰.۵ گره n0 شروع به ارسال ترافیک برای گره n1 می کند. در ثانیه ۴/۵ ارسال ترافیک موقوف می شود. شکل زیر ارسال ترافیک از گره n0 به گره n1 را نشان می دهد.




می توانید با کلیک روی لینک اطلاعات آماری را بدست آورید . همین طور می توانید اندازه بسته ها و زمان بین ارسال بسته ها^۱ را تغییر داده و حاصل تغییرات را مشاهده کنید.

۱-۲-۶-۱- مشخص کردن جریانهای ترافیک و پایش کردن لینکها

در این بخش از یک توپولوژی با ۴ گره استفاده خواهیم کرد . دو گره به عنوان منبع ترافیک عمل می کنند . یک گره دریافت کننده ترافیک می باشد . و یک گره فقط جریانهای ترافیک را به مقصد می فرستد. برای مشخص کردن جریانهای ترافیک و مونیتور کردن صف مربوط به لینک روشهایی ارائه خواهد شد

^۱ Interval Time

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

۱-۶-۳- تعریف توپولوژی

حال برای تعریف گره ها خطوط زیر را اضافه کنید:

```
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

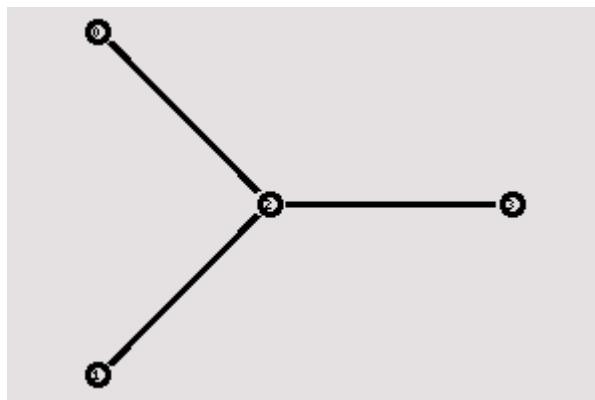
برای تعریف لینک بین گره ها کدهای زیر را بیفزایید:

```
$ns duplex-link $n0 $n2 1mb 10ms DropTail
$ns duplex-link $n1 $n2 1mb 10ms DropTail
$ns duplex-link $n3 $n2 1mb 10ms DropTail
```

اکنون فایل را ذخیره و اجرا کنید . البته ممکن است شکل آن دلخواه نباشد . می توانید دکمه re-layout را بزنید تا شکل آن بهتر شود . اما بهتر است کنترل بیشتری روی آن داشته باشید . برای این منظور از پیکربندی مستقیم ها می توان استفاده کرد.

```
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
```

با افزودن این دستورات ها شکل توپولوژی در nam بصورت زیر خواهد شد.




در دستورهای بالا جهت لینک ها را با right, left, up, down و یا ترکیبی از اینها می توان مشخص کرد.

۱-۶-۴- تعریف منابع ترافیک و گیرنده های ترافیک

گره های n0 و n1 را بعنوان منابع ترافیک CBR و گره n3 را بعنوان گیرنده ترافیک تعریف می کنیم.
گره n2 تنها کار ارسال را برای جریانهای ترافیک انجام می دهد.

```
#create a UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
#create a CBR traffic source and attach it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetize_500
```


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```

$nbr0 set interval _0.005
$nbr0 attach-gent $udp0
#create a UDP agent and attach it to noder n1
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
#create a CBR traffic source and attach it to udp1
set cbr1 [new Application/Traffic/CBR]
$nbr1 set packetize _500
$nbr1 set interval _0.005
$nbr1 attach-gent $udp1
#create a null agent for traffic sink n3
set null0 [new Agent/Null]
$ns attach-agent $n3 $null0

```

حال باید اتصال عاملها برقرار گردد

```

$ns connect $udp0 $null0
$ns connect $udp1 $null0

```

می خواهیم منبع ترافیک cbr0 در ثانیه ۰.۵ شروع به ارسال ترافیک کرده و در ثانیه ۴ ارسال ترافیک را متوقف کند و منبع ترافیک cbr1 در ثانیه 1 شروع به ارسال ترافیک کرده و در ثانیه ۴/۵ ارسال ترافیک را متوقف کند. بنابراین به صورت زیر عمل میکنیم.

```

$ns at 0.5 $nbr0 start
$ns at 1.0 $nbr1 start
$ns at 4.0 $nbr0 stop
$ns at 4.5 $nbr1 stop

```

با اجرای این اسکریپت ، متوجه می شوید که مجموع دو جریان ترافیک ، بیش از ظرفیت خط بین n2 و n3 بوده و سبب حذف شدن برخی از بسته ها می شود. یک محاسبه ساده این مطلب را تایید می کند. اندازه هر بسته ۵۰۰ بایت بوده و در هر ثانیه ۲۰۰ بسته ارسال می شود. پس هر جریان ترافیک پهنای باند ۰.۸ مگا بیت در ثانیه نیاز دارد. و خط ۱ مگابیت در ثانیه بین n2 و n3 جوابگوی ترافیک ۱/۶ مگا بیت در ثانیه ارسالی برای آن نیست.

پس بخشی از بسته ها حذف می شوند. اما کدام بسته ها ؟ چون رنگ تمام بسته ها مشکی است تمایز بین آنها امکان پذیر نمی باشد. برای ایجاد تمایز بین جریانهای ترافیک در nam، از دستورات پیکربندی خاصی استفاده می شود. این دستورات در بخش بعد توضیح داده شده اند.

۱-۶-۱-۵- مشخص کردن جریانهای ترافیک


دو خط زیر را به اسکریپت خود اضافه کنید:

```

$ns color 1 blue
$ns color 2 red

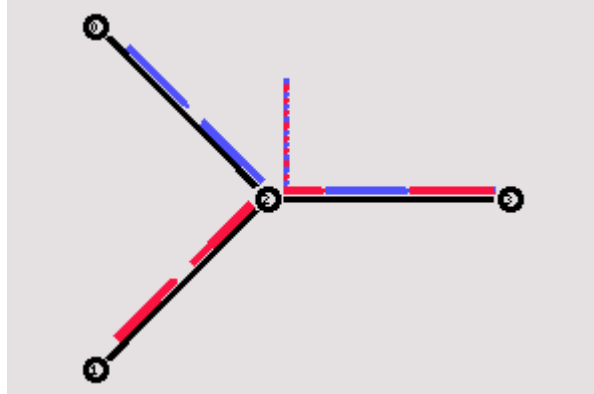
```

اگر اسکریپت را دوباره اجرا کنید، متوجه می شوید که جریان ترافیک n0 آبی رنگ و جریان ترافیک n1 قرمز رنگ می باشد. اگر لینک بین گره های n2 و n3 را مشاهده کنید خواهید دید که بعد از مدتی نحوه حذف بستهها زیاد عادلانه نیست.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۶-۱-۶-۱- مونیطور کردن یک لینک

برای مونیطور کردن صف مربوط به لینک بین $n2$ و $n3$ دستور زیر را به اسکریپت خود بیفزایید.
`$ns duplex-link-op $n2 $n3 queuePos 0.5`
 پس از چند لحظه تصویری مشابه تصویر زیر مشاهده خواهید کرد.



شکل ۳۷: مونیطور کردن یک لینک

اکنون می توانید قرار گرفتن بسته ها را در صف ببینید و بسته هایی که حذف می شوند را مشاهده کنید همانگونه که قبلا گفتیم نحوه حذف بسته ها عادلانه نیست.
 چون صف از نوع DropTail است بنا براین نمی توان انتظار داشت که عدالت را در حذف بسته ها رعایت کند. برای اینکه با هر دو نوع بسته به صورتی یکسان برخورد شود می توانید از صف SFQ استفاده کنید.
 به این منظور تعریف لینک بین $n2$ و $n3$ را به این صورت تغییر دهید.
`$ns duplex-link $n3 $n2 1Mb 10ms SFQ`
 صف بندی باید اکنون عادلانه باشد. بسته های آبی و قرمز به یک میزان حذف میشوند..

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

معرفی شبکه های VANET

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۲-۱- مقدمه

۲-۲- بررسی شبکه های VANET

شبکه های VANET نوعی از شبکه های Ad_hoc هستند که ارتباط بین وسایل نقلیه مجاور، همچنین بین وسایل نقلیه و تجهیزات ثابتی که معمولاً کنار جاده ها نصب می شوند را فراهم می آورند، هدف اصلی در شبکه های vanet ایجاد راحتی و امنیت بیشتر برای مسافران می باشد. به این منظور یک وسیله الکترونیکی در هر وسیله نقلیه نصب می شود که اتصال به شبکه Ad_hoc را برای مسافران فراهم می آورد، به این ترتیب هر وسیله نقلیه ای که با تجهیزات vanet مجهز شده است بعنوان یک نود در شبکه Ad_hoc عمل می کند و می تواند پیغام های دیگران را از طریق شبکه بی سیم دریافت و یا به نودهای دیگر پیغام ارسال نماید، این پیغام ها بیشتر برای اهداف امنیتی و کنترل ترافیک به کار می روند از جمله می توان به موارد زیر اشاره نمود :

- هشدار تصادف
- اعلان علائم جاده ای
- مشاهده ترافیک
- پرداخت عوارض راهداری
- پرداخت هزینه پارکینگ
- تفریحات و سرگرمی
- سایر موارد

این پیغام ها برای رانندگان ابزارهای مناسبی جهت تصمیم گیری در خصوص مسیر مناسب ارائه می دهند. علاوه بر این امکانات چند رسانه ای و اینترنت نیز برای مسافران تعبیه شده است. پرداخت عوارض راهداری و هزینه پارکینگ نیز از سایر خدمات این نوع شبکه ها است. امروزه کارخانه های بزرگ خودروسازی از جمله تویوتا و BMW پروژه های مختلفی را در خصوص این نوع شبکه ها و تجهیز خودروهای خود به قابلیت های شبکه های VANET در دستور کار دارند. شبکه های بیسیم تک کاره وسایل نقلیه (VANETs) یک گروه فوق العاده چالشی از شبکه های بیسیم تک کاره تغییرپذیر (MANETs) هستند که در حال حاضر توجه گسترده تحقیقات در زمینه شبکه سازی بیسیم و همچنین صنایع اتومبیل سازی را به خود جلب کرده اند. برقراری ارتباط در MANETs به برنامه های کاربردی متنوع و پراکنده در میان گره های سیار در محیط های فاقد ساختار زیربنایی کمک میکند. به علت داشتن قابلیت تغییرپذیری نسبتاً زیاد ، برقراری ارتباط در VANETs چالش های قویتری را نسبت به ارتباط های دیگر در MANETs عمومی به نمایش

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

میگذارد. محیطهای فاقد ساختار زیربنایی و توپولوژی شبکه ای با پویایی بالا موجب تقسیم بندی مداوم شبکه ها میشود. شبکه های vanet را می توان جزئی از سیستم های انتقال هوشمند^۱ ITS مطرح کرد همکاری دولتها، گروههای استاندارد سازی و تولیدکنندگان سراسر جهان پیشرفت سیستمهای انتقال هوشمند (ITS) را تسریع میکند. موفقیت بالقوه تحقیقات VANETs، به عنوان بخش ضروری تحقیقات ITS، وابسته به توسعه سیستم ارتباط وسایل نقلیه است که توزیع مناسب، پایدار و مقرون به صرفه اطلاعات را در جهت بهره گیری از امنیت و آسایش "در راه انتقال" امکانپذیر میسازد.

در میان برنامه های کاربردی ارتباطی گوناگون در VANETs، طیف گسترده ای از برنامه های مهم وجود دارد که امنیت ترافیک، کنترل ترافیک و برنامه های کاربردی وسایل نقلیه فاقد راننده را که نیازمند ارتباط محدود به زمان در شبکه های بیسیم تک کاره هستند، شامل میشود. این برنامه های توزیع شده دارای محدودیت زمانی را ارتباط همزمان مینامیم.

همانطور که در بالا بررسی شد، گره های بسیار پویا و سرور فاقد ایستگاه مرکزی موجب بروز تصادم در رسانه بیسیم در ارتباط VANETs میشوند. در رسانه مورد نظر، تاخیر و مفقود شدن بسته ها مرتباً اتفاق می افتد. ارتباط همزمان به راحتی در این سناریوها دچار بی حاصلی میشود. بنابراین، توسعه یک سری سازوکارهای موثر در تضمین اتمام این ارتباطهای حساس به زمان، در یک بازه زمانی محدود ضروری است شبکه های VANET در بسیاری از موارد شبیه شبکه های MANET هستند از جمله می توان به موارد زیر اشاره نمود:

- دامنه کوتاه انتقال امواج رادیویی^۲
- پهنای باند کوتاه^۳
- انتشار امواج در جهت مناسب^۴
- ظرفیت ذخیره سازی محدود^۵

اما در جزئیات با این نوع شبکه ها تفاوت هایی دارند از جمله این که حرکت در شبکه های VANET ساختاریافته است زیرا معمولاً حرکت خودروها در مسیرهای مشخصی در طول جاده ها و خیابان ها انجام می شود. بطور کلی میتوان برقراری ارتباط در VANETs ویژگیهای چالشی به خصوصی را در بر میگیرد:

- تغییر سریع توپولوژی^۶

^۱ Intelligent Transportation System


^۲ Short radio transmission range

^۳ Low bandwidth

^۴ Omni directional broadcast

^۵ Limited storage capacity

^۶ Rapid topology changes

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

- طبقه بندی مداوم شبکه^۱
 - شبکه میانبر کوچک و موثر^۲
 - افزونگی محدود به زمان و عملکرد^۳
 - قابلیت پیش بینی موقعیت^۴
 - قدرت به نسبت کافی^۵
- قابلیت پیش بینی موقعیت و قدرت به نسبت کافی را میتوان در جهت پشتیبانی ارتباط میان وسایل نقلیه و میان وسیله نقلیه و حاشیه جاده به کار گرفت، در حالی که تغییر سریع توپولوژی، طبقه بندی مداوم شبکه، شبکه میانبر کوچک و موثر و افزونگی محدود در زمان و عملکرد مشکلات ارتباط را در VANETs تشدید میکند.
- در سناریوهای VANETs سه چالش تعیین کننده وجود دارد که نقشی اساسی در دستیابی به ارتباط پایدار و موثر را ایفا میکنند که عبارتند از:
- چگونگی بهره گیری موثر از پهنای باند کوتاه
 - چگونگی حفظ و نگهداری توپولوژی قطعه بندی شده به صورت پویا^۶
 - چگونگی دستیابی به حداقل تاخیر در انتقال همزمان اطلاعات در موقعیتهای گوناگون

۲-۳- تکنولوژی ها و پروتکل های شبکه های VANET

شبکه های VANET تکنولوژی های مختلف بیسیم مانند DSRC^۷ را که نوعی WIFI است همچنین تکنولوژی سلولی و WIMAX را به کار می گیرد. DSRC کانال های مخابراتی با برد کوتاه و متوسط به صورت یک طرفه و دو طرفه است که مشخصا برای اهداف متحرک مطابق با یک سری استانداردها و پروتکل ها طراحی شده است.

^۱ Frequent network partition

^۲ Small effective network diameter

^۳ Limited redundancy in time and in function

^۴ Position predictability

^۵ Relatively sufficient power

^۶ Dynamically fragmented topology

^۷ Dedicated Short Range Communication

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

پروتکل های بی سیم باند کوتاه دیگر شامل IEEE 802.11 ، Blue tooth و CALM در این نوع شبکه ها قابل استفاده می باشند.

بکارگیری پروتکل های مسیریابی مختلف مانند ¹ DSR و ² DSDV در شبکه های vanet مورد بررسی قرار دارد ولی به نظر می رسد پروتکل های مسیریابی شبکه های ad_hoc مانند ³ AODV و DYMO در این نوع شبکه ها کارایی بیشتری دارند.


همچنین شبکه های VANET را می توان بعنوان جزئی از سیستم های انتقال هوشمند ⁴ مطرح کرد.

¹ Dynamic Source Routing

² Destination Sequence Distance Vector


³ Ad hoc On-demand Distance Vector

⁴ ITS: Intelligent Transporatation Systems

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

شبیه سازی شبکه های VANET

در NS2

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۵-۱- مقدمه

پروتکل های مختلف بی سیم در NS2 تعریف شده است این شبیه ساز امکان تنظیم بسیاری از پارامترهای مختلف را در توپولوژی های بیسیم در اختیار قرار می دهد. برای مثال می توان مدل انتشاری محیطی که شبیه سازی در آن صورت می گیرد را از بین مدل های افت فضای آزاد^۱ و دوشعاعه^۲ به همراه سایه^۳ انتخاب نمود. تنها پترن آنتن تعریف شده برای NS، Omni-Directional است. با توجه به ماهیت Open-Source بودن NS، انواع دیگر آنتن و کانال قابل تعریف است.

همچنین انواع مختلفی از پروتکل های Routing برای شبکه Wireless تعریف شده است که از آن جمله می توان به (DSDV) Destination Sequence Distance Vector، (DSR) Dynamic Source Routing، (TORA) Temporally ordered Routing Algorithm و (AODV) Adhoc On-demand Distance Vector اشاره نمود.

یکی از ویژگی های منحصر به فرد NS، امکان ایجاد تحرک^۴ در شبیه سازی شبکه های بیسیم است. بدین معنی که با تعریف موقعیت مکانی اولیه Station ها، سرعت و در نهایت بردار جهت حرکت آنها، می توان در حین شبیه سازی، گیرنده و فرستنده ها را جابجا نمود. NS خود با در نظر گرفتن تاخیر ذاتی مسیر بین نود ها و مدل افت محیط، توان دریافتی (میزان خطا) و سایر پارامتر های شبیه سازی را لحظه به لحظه محاسبه می کند. می توان برای media یک مدل خطا نیز در نظر گرفت (Error Model). علاوه بر این امکان تعریف حرکت تصادفی (بصورت آماری) نیز برای Station ها وجود دارد.

پروتکل DCF^۵ لایه MAC برای IEEE 802.11 در NS2 پیاده سازی شده است. این پروتکل برای حالت Unicast از الگوی RTS/CTS/DATA/ACK استفاده می کند (حالت معمولی) و برای حالت Broadcast بسته را به همه گیرنده ها ارسال می کند. علاوه بر این پروتکل، پروتکلی بنام TDMA-MAC نیز برای لایه MAC طراحی شده است که در دست توسعه می باشد. در این روش، پروتکل، بازه های زمانی (Time-slot) متفاوتی را در یک فریم برای ارسال و دریافت داده های نود ها استفاده می کند.

برای شبیه سازی شبکه های مرکب (Wired+Wireless) مجموعه رویه هایی به NS اضافه گردید که از آن جمله نود "BaseStationNode" است که مانند یک Gateway بین شبکه Wired و Wireless عمل می کند. با تعریف این شیء هر مجموعه از Station های بیسیم در یک Domain با یک شماره مشخصه یکتا قرار می گیرند و BaseStationNode آن حوزه، مسوول رساندن بسته ها به مقاصد Wireless می باشد. پس از تعریف این شیء، امکان پیاده سازی پروتکل های مسیر یابی در توپولوژی های مختلط (سیم و بیسیم)

¹Free Space

²Two-Ray

³Shadowing

⁴Mobility

⁵Distributed coordination function (DCF)

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

میسر شده و نیز پروتکل هایی مانند MobileIP برای شبکه Wireless تعمیم داده شد. (پروتکل MobileIP توسط SUN Micro Systems ابتدا برای توپولوژی های Wired تعریف و پیاده سازی شده بود) در مجموع امکانات NS برای شبیه سازی Wireless محدود می باشد. اگرچه بعضی ویژگی های آن همانند قابلیت تحرک کاربران منحصر به فرد می باشد، اما همه حالات ممکن، (مثلاً Roaming) در آن لحاظ نشده است. علاوه بر این، NS انواع اندکی از ترافیک های از پیش تعریف شده (مانند CBR، VBR، FTP و ...) را در اختیار قرار می دهد و با توجه به رابط کاربری مبتنی بر text آن، ایجاد توپولوژی یک شبکه بزرگ و شبیه سازی ترافیک نزدیک به واقعیت روی آن بسیار پیچیده و وقتگیر خواهد بود.

ایجاد مدل حرکتی نودها با جزئیات کامل که شامل حرکت خودروها در مسیرهای خاص و در جهات رفت و برگشت، سرعت حرکت خودروها و غیره می شود به راحتی امکان پذیر نمی باشد برای حل این مشکل معمولاً از شبیه سازهای کمکی دیگری مانند SUMO و MOVE استفاده می شود. در این بخش از گزارش نمونه های شبیه سازی شده ای از VANET ارائه شده است در نمونه ۱ و ۲ این شبیه سازی ها فقط با استفاده از قابلیت های شبیه ساز NS2 انجام شده است و در نمونه ۳ مدل حرکتی نودها با استفاده از شبیه سازهای SUMO، MOVE انجام شده است.

۵-۲- نمونه شبیه سازی شده ۱

در این بخش یک برنامه tcl و نحوه خروجی گرفتن از برنامه ارائه شده است.


۵-۲-۱- طرز اضافه کردن یک پروتکل جدید به NS-2

برای استفاده از هر پروتکل در برنامه TCL باید به مشخصات آن پروتکل کاملاً اطلاع داشته باشید. برای اینکه بتوان در برنامه TCL پارامترهای مختلفی که در یک پروتکل تعریف گردیده مقداردهی کرد. بایستی ابتدا به پوشه protocol.cc در برنامه NS-2 نگاهی بیندازیم این پارامترها با دستور bind به اصطلاح به برنامه tcl چسبانده شود. در زیر یک مثال آورده شده است.

protocol.cc
protocol:protocol () { bind("DelayCount_", &DelayCount); }
simulation.tcl
Agent/Protocol set DelayCount_ 1


حال اگر میخواهید پروتکلی را که تا کنون در برنامه NS شما وجود نداشته به آن اضافه کنید. مجبورید بعضی از فایل های NS را تغییر دهید. در این قسمت کارهای مورد نیاز را برای تغییر فایلها را شرح می دهد.


common/packet.h
class p_info { public:

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
<pre>p_info() { name_[PT_TCP]= "tcp"; name_[PT_PROTOCOL] = "protocol";</pre>		
<pre>tcl/lib/ns-default.tcl</pre>		
<pre>Agent/Protocol set packetSize_ 32</pre>		
<pre>tcl/lib/ns-packet.tcl</pre>		
<pre>foreach prot { AODV Protocol</pre>		
<pre>Makefile</pre>		
<pre>OBJ_CC = \ random.o rng.o ranvar.o misc.o timer-handler.o \ ... protocol/protocol.o \$(OBJ_STL)</pre>		

گاهی اوقات مجبور هستیم که فایل های cmu-trace.h و cmu-trace.cc را نیز تغییر دهیم تا پروتکل شبیه سازی بیسیم فایلها را تولید می کند.

cmu-trace.cc
<pre>#include <protocol/protocol.h> // command void CMUTrace::format_protocol(Packet *p, int offset) { // packet } // another syntax void CMUTrace::format(Packet* p, const char *why) hdr_cmn *ch = HDR_CMN(p); int offset = 0; /* * Log the MAC Header */ format_mac(p, why, offset); if (namChan_) nam_format(p, offset); offset = strlen(wrk_); switch(ch->ptype()) { case PT_MAC: break; case PT_ARP: format_arp(p, offset); break;</pre>

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre> default: format_ip(p, offset); offset = strlen(wrk_); switch(ch->ptype()) { case PT_AODV: format_aodv(p, offset); break; case PT_TORA: format_tora(p, offset); break; case PT_IMEP: format_imep(p, offset); break; case PT_DSR: format_dsr(p, offset); break; case PT_MESSAGE: case PT_UDP: format_msg(p, offset); break; case PT_TCP: case PT_ACK: format_tcp(p, offset); break; case PT_CBR: format_rtp(p, offset); break; case PT_PROTOCOL: format_protocol(p, offset); break; default: fprintf(stderr, "%s - invalid packet type (%s).\n", __PRETTY_FUNCTION__, packet_info.name(ch->ptype())); exit(1); } } } } } </pre>		
<pre> cmu-trace.h class CMUTrace : public Trace { public: CMUTrace(const char *s, char t); void recv(Packet *p, Handler *h); void recv(Packet *p, const char* why); private: char tracename[MAX_ID_LEN + 1]; int nodeColor[MAX_NODE]; </pre>		

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre> int tracetype; MobileNode *node_; int newtrace_; int initialized() { return node_ && 1; } int node_energy(); int command(int argc, const char*const* argv); void format(Packet *p, const char *why); void nam_format(Packet *p, int offset); void format_mac(Packet *p, const char *why, int offset); void format_ip(Packet *p, int offset); void format_arp(Packet *p, int offset); void format_dsr(Packet *p, int offset); void format_msg(Packet *p, int offset); void format_tcp(Packet *p, int offset); void format_rtp(Packet *p, int offset); void format_tora(Packet *p, int offset); void format_imep(Packet *p, int offset); void format_aodv(Packet *p, int offset); void format_protocol(Packet *p, int offset); }; </pre>		


همانطور که در بخش های قبلی اشاره شد روش متداول برای تولید برنامه tcl برای شبکه های vanet استفاده از نرم افزار move و sumo می باشد. در ادامه به بیان چگونگی تولید برنامه Tcl در شبیه ساز ns2 اشاره می شود. البته در یک روش دیگر می توان از نرم افزار مطلب نیز استفاده کرد. برای این کار حرکت گره ها به صورت تصادفی در مطلب تولید می شود پس از آن به نرم افزار ns2 بار می شود.

در ابتدا برنامه TCL باید ابتدا مشخصات مختلف به صورت زیر وارد شود. مشخصات آمده در جدول زیر برای یک شبکه بیسیم می باشد. اکثر برنامه های TCL از استاندارد زیر استفاده می کنند. از دستور SET برای تعیین مشخصات مختلف استفاده می شود. برای مثال در آخر پارامتر nn مقدار ۲ را می گیرد که در ادامه از آن استفاده می شود. یا در خط زیر

```
set val(rp) DSDV ;# ad-hoc routing protocol
```

rp مقدار DSDV را می گیرد. در کل برنامه هر وقت rp را می نویسیم مقدار آن برابر با DSDV است.

Define options
<pre> set val(chan) Channel/WirelessChannel ;# channel type set val(prop) Propagation/TwoRayGround ;# radio-propagation model set val(ant) Antenna/OmniAntenna ;# Antenna type set val(ll) LL ;# Link layer type set val(ifq) Queue/DropTail/PriQueue ;# Interface queue type set val(ifqlen) 50 ;# max packet in ifq set val(netif) Phy/WirelessPhy ;# network interface type set val(mac) Mac/802_11 ;# MAC type set val(rp) DSDV ;# ad-hoc routing protocol set val(nn) 2 ;# number of mobilenodes </pre>

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---


در برنامه Tcl مورد نظر باید ابتدا ns_node-config تعریف شود. در ادامه یک فایل ns-node-config ساده آورده شده است.

<pre>\$ns_node-config -addressingType flat or hierarchical or expanded -adhocRouting DSDV or DSR or TORA -llType LL -macType Mac/802_11 -propType "Propagation/TwoRayGround" -ifqType "Queue/DropTail/PriQueue" -ifqLen 50 -phyType "Phy/WirelessPhy" -antType "Antenna/OmniAntenna" -channelType "Channel/WirelessChannel" -topoInstance \$topo -energyModel "EnergyModel" -initialEnergy (in Joules) -rxPower (in W) -txPower (in W) -agentTrace ON or OFF -routerTrace ON or OFF -macTrace ON or OFF -movementTrace ON or OFF</pre>

فرمت بالا حالت استاندارد است. در غیر این صورت خطا خواهد داد. فاصله بین خطوط اصلا نباید آورده شود. بعضی اوقات ممکن برنامه بدون هیچ دلیلی خطا دهد برای این کار بهتر است از یک برنامه پیش فرض استاندارد استفاده شود. تا حد امکان از copy/paste خودداری کنید. زیرا بعضی اوقات برنامه خطا می دهد بدون اینکه دلیل آن را متوجه باشید. فیلد addressType نشان دهنده همبندی شبکه است. بیان گر این است که شبکه به صورت سلسله مراتبی یا معمولی باشد. جلوی addressType ، یکی از پرامترها نوشته شود برای مثال flat انتخاب شود. adhocRouting نشان دهنده نوع مسیریابی است که باید استفاده شود. انواع مسیریابی های مختلف را در فایل Lib می توانید مشاهده کنید.

شبکه های vanet براساس استاندارد 802.11 می باشد در نتیجه باید پروتکل لایه mac شما 802.11 انتخاب شود. در غیر این صورت اگر پروتکل mac دیگری تعریف کرده اید آن را می توانید ذکر کنید. بقیه قسمتها همانند نوع آنتن و انرژی مصرف طبق فرمت استاندارد باید ذکر شود. برای موبایل کردن گره ها قسمتهای قرمز رنگ باید حتما ON شود.

<p align="center">ایجاد یک توپولوژی جدید</p> <pre>set ns_ [new Simulator] set tracefd [open simple.tr w] \$ns_ trace-all \$tracefd set topo [new Topography] Stopo load_flatgrid 500 500</pre>

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
create-god \$val(nn)		

در هر برنامه tcl باید خط [set ns_ [new simulator]] آورده شود. در خط دوم به فایل خروجی برنامه ns2 اشاره می شود. فایل خروجی با نام simple و به فرمت tr می باشد.

خط سوم نشان دهنده این است که همه مشخصات اجرای برنامه در فایل خروجی نشان دهد. در خط چهارم یک توپولوژی جدید ایجاد می شود و در خط پنجم نیز مساحت شبیه سازی را نشان می دهد که در این مثال 500x500 می باشد.

در خط آخر نیز نوع توپولوژی را نشان می دهد که به صورت خوشه بندی است و تعداد سر خوشه ها برابر با ۲ است. همانطور که قبلا بیان شد مقدار nn برابر با ۲ در نظر گرفته شده است.

در ادامه به بررسی گره های موبایل اشاره می شود.

متحرک کردن گره ها
<pre> for {set i 0} {\$i < \$val(nn)} {incr i} { set node_(\$i) [\$ns_ node] \$node_(\$i) random-motion 0 ; motion } # disable random ## Provide initial (X,Y, for now Z=0) co-ordinates ## for node_(0) and node_(1) \$node_(0) set X_ 5.0 \$node_(0) set Y_ 2.0 \$node_(0) set Z_ 0.0 \$node_(1) set X_ 390.0 \$node_(1) set Y_ 385.0 \$node_(1) set Z_ 0.0 node_(0) ## Node_(1) starts to move towards 15.0" \$ns_ at 50.0 "\$node_(1) setdest 25.0 20.0 \$ns_ at 10.0 "\$node_(0) setdest 20.0 18.0 1.0" node_(0) # Node_(1) then starts to move away from \$ns_ at 100.0 "\$node_(1) setdest 490.0 480.0 15.0" \$ns_ at 50.0 "\$node_(1) setdest 25.0 20.0 15.0" </pre>

در این قسمت دو گره موبایل در نظر گرفته شده است که در حلقه for برای هر کدام از گره ها یک اسم در نظر گرفته می شود. حلقه For اغلب برای تعداد بیشتر گره ها استفاده می شود. برای دو حلقه می توانید از دستور set استفاده کنید. پارامترهای دیگر برای مشخص کردن جایگاه اولیه گره ها استفاده می شود که با پارامترهای x و y و z مشخص شده است. در ادامه به بررسی مهم ترین دستور که برای شبکه های vanet استفاده می شود پرداخته می شود.

برای تولید متحرک بودن یک گره، یک دستور اجرایی setdest استفاده می شود. فرمت کلی دستور setdest به صورت زیر می باشد که در ادامه به آن اشاره شده است.

```
setdest -v 1 -n 20 -p 2.0 -M 10.0 -t 200 -x 500 -y 500
```

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

```
setdest -v 2 -n 20 -s 1 -m 1 -M 10.0 -t 200 -P 1 -p 2.0 -x 500 -y 500
```

در جدول زیر فایل های خروجی آن نشان داده شده است.


	Function	Range	Default	Ver 1	Ver 2
-M	maximum speed	≥ 0	0.0	Yes	Yes
-m	minimum speed	[0, maximum speed]	0.0	No, always zero	Yes
-n	number of nodes	1,2,...	0	Yes	Yes
-p	pause type	1 (constant), 2 (uniform)	1	No	Yes
-P	pause time	≥ 0	0.0	Yes	Yes
-s	speed type	1 (uniform), 2 (normal)	1	No	Yes
-t	maximum time	≥ 0	0.0	Yes	Yes
-v	version	1 (1999 CMU), 2 (2003 UM)	1	Yes	Yes
-x	width of space	≥ 0	0.0	Yes	Yes
-y	height of space	≥ 0	0.0	Yes	Yes

برای اجرای دستورات مربوط به setdest به پوشه زیر بروید. در مثال زیر فرض بر این است شبیه ساز ns2 در پوشه زیر نصب شده است. طبق آدرس زیر به فایل setdest بروید.

en\setdest E:\cgywin\home\Administrator\ns-allinone-2.28\ns-2.28\indep-utils\cmu-scen-
بعد از آن مرحله دستور زیر را تایپ کنید.


```
setdest -v 1 -n 20 -p 2.0 -M 10.0 -t 200 -x 500 -y 500 > scen-20_v1.tcl
setdest -v 2 -n 20 -s 1 -m 1 -M 10.0 -t 200 -P 1 -p 2.0 -x 500 -y 500 >
scen-20_v2.tcl
```

بعد از این کار دو فایل خروجی scen-20_v1.tcl و scen-20_v2.tcl را می دهد که می توانید برای متحرک بودن گره ها استفاده کنید. در ادامه به برنامه tcl و نحوه خروجی گرفتن از برنامه پرداخته می شود.

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۵-۲-۲- برنامه tcl با چهار گره موبایل


```
#
=====
# Define options
#
=====
set val(chan) Channel/WirelessChannel ;# channel type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
set val(mac) Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 4 ;# number of mobilenodes
set val(rp) AODV ;# routing protocol
set val(x) 500 ;# X dimension of the topography in meters
set val(y) 500 ;# Y dimension of the topography in meters
set val(time) 2.0 ;# Simulation time in seconds
set data_interval 0.2 ;# CBR Traffic Interval
set start_time 0.5 ; # traffic start time
proc finish {} {
global ns_ tracefile namfile
$ns_ flush-trace
close $tracefile
close $namfile
exit 0
}
#
=====
# Main Program
#
=====
#
# Initialize Global Variables
#
set ns_ [new Simulator]
# Trace Files
set tracefile [open out.tr w]
$ns_ use-newtrace
$ns_ trace-all $tracefile
set namfile [open out.nam w]
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

$ns_ namtrace-all-wireless $namfile $val(x) $val(y)
# set up topography object
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
#
# Create God
#
create-god $val(nn)
set chan [new $val(chan)]
# configure node
$ns_ node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON \
-movementTrace ON \
-channel $chan
for {set i 0} {$i < $val(nn)} {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;# disable random motion
$ns_ initial_node_pos $node_($i) 20
}
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes
# and produce some simple node movements
$node_(0) set X_ 190
$node_(0) set Y_ 190
$node_(0) set Z_ 0.0
$node_(1) set X_ 100
$node_(1) set Y_ 100
$node_(1) set Z_ 0.0
$node_(2) set X_ 280
$node_(2) set Y_ 100
$node_(2) set Z_ 0.0
$node_(3) set X_ 50
$node_(3) set Y_ 230
$node_(3) set Z_ 0.0
# Initial Movement
for {set i 0} {$i < $val(nn)} {incr i} {

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

$ns_ at 0.0 "$node_($i) setdest 10.0 10.0 0.0"
}
# Null Agent to receive Packets for Node 0
set null_(0) [new Agent/Null]
>null_(0) set fid_ 1
$ns_ attach-agent $node_(0) $null_(0)
# Null Agent to receive Packets for Node 3
set null_(1) [new Agent/Null]
>null_(1) set fid_ 2
$ns_ attach-agent $node_(3) $null_(1)
# Data Source , Nodes
set udp_(0) [new Agent/UDP]
$udp_(0) set fid_ 0
$ns_ attach-agent $node_(1) $udp_(0)
# Constant Bit rate traffic genrator
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 1024
$cbr_(0) set interval_ $data_interval
$cbr_(0) set random_ 0
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at $start_time "$cbr_(0) start"
set udp_(1) [new Agent/UDP]
$udp_(1) set fid_ 0
$ns_ attach-agent $node_(2) $udp_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 1024
$cbr_(1) set interval_ $data_interval
$cbr_(1) set random_ 0
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at $start_time "$cbr_(1) start"
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn)} {incr i} {
$ns_ at $val(time) "$node_($i) reset";
}
$ns_ at $val(time) "finish"
$ns_ at [expr $val(time) + 0.01] "puts \"NS EXITING...\"; $ns_ halt"
puts "Starting Simulation..."
$ns_ run

```

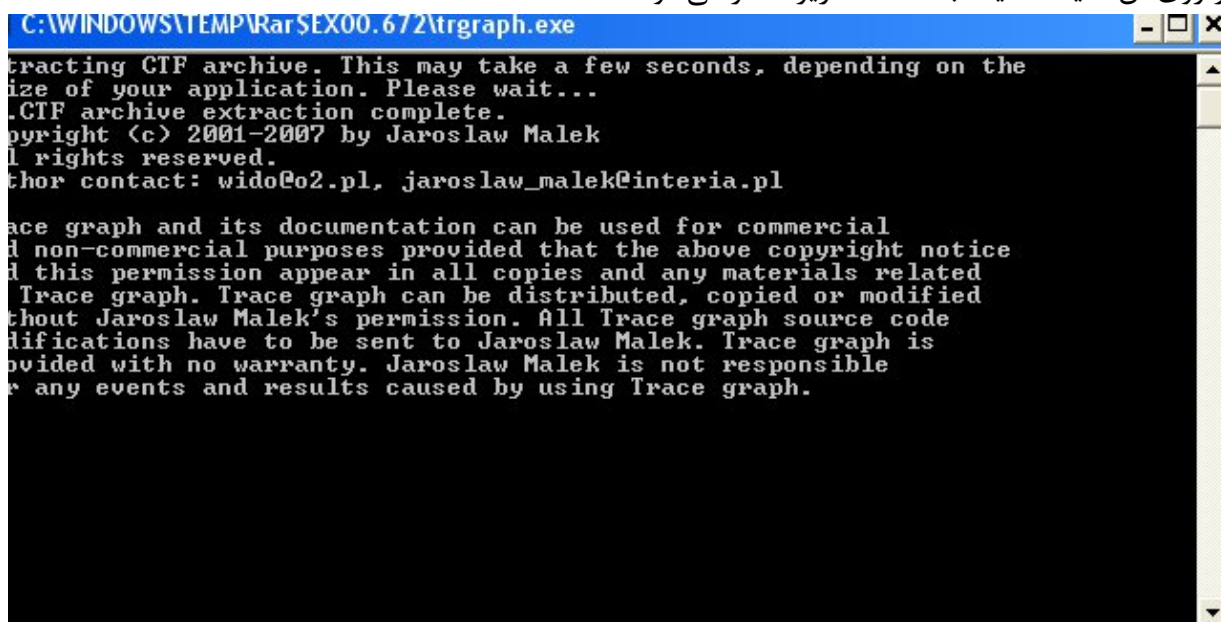
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

شبیه ساز ns2 باید یا برنامه مطلب نصب شود یا کتابخانه MCRInstaller.msi نصب شود. برای دانلود کتابخانه MCRInstaller.msi می توانید از سایت زیر استفاده کنید.

<http://www.tracegraph.com/>

پس از آن فایل tracegraph را از سایت بالا دانلود کنید. همانطور که قبلا اشاره شد خروجی برنامه Tcl یک فایل tr است که ورودی برنامه xgraph می باشد.

برنامه xgraph را از حالت زیپ شده خارج کنید. در داخل آن یک فایل به نام trgraph.exe وجود دارد بر روی آن کلیک کنید. ابتدا صفحه زیر ظاهر می شود.



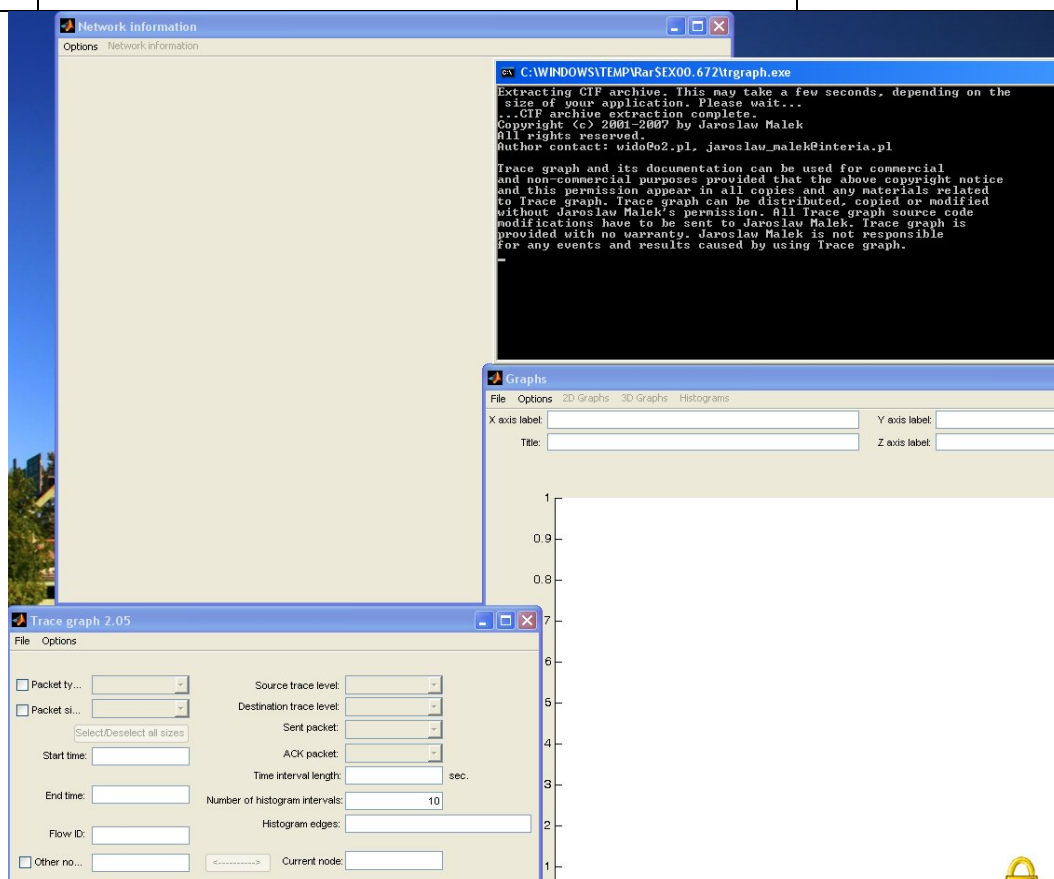
```

C:\WINDOWS\TEMP\Rar$EX00.672\trgraph.exe
Extracting CTF archive. This may take a few seconds, depending on the
size of your application. Please wait...
CTF archive extraction complete.
Copyright (c) 2001-2007 by Jaroslav Malek
All rights reserved.
Author contact: wido@o2.pl, jaroslav_malek@interia.pl

Trace graph and its documentation can be used for commercial
and non-commercial purposes provided that the above copyright notice
and this permission appear in all copies and any materials related
to Trace graph. Trace graph can be distributed, copied or modified
without Jaroslav Malek's permission. All Trace graph source code
modifications have to be sent to Jaroslav Malek. Trace graph is
provided with no warranty. Jaroslav Malek is not responsible
for any events and results caused by using Trace graph.

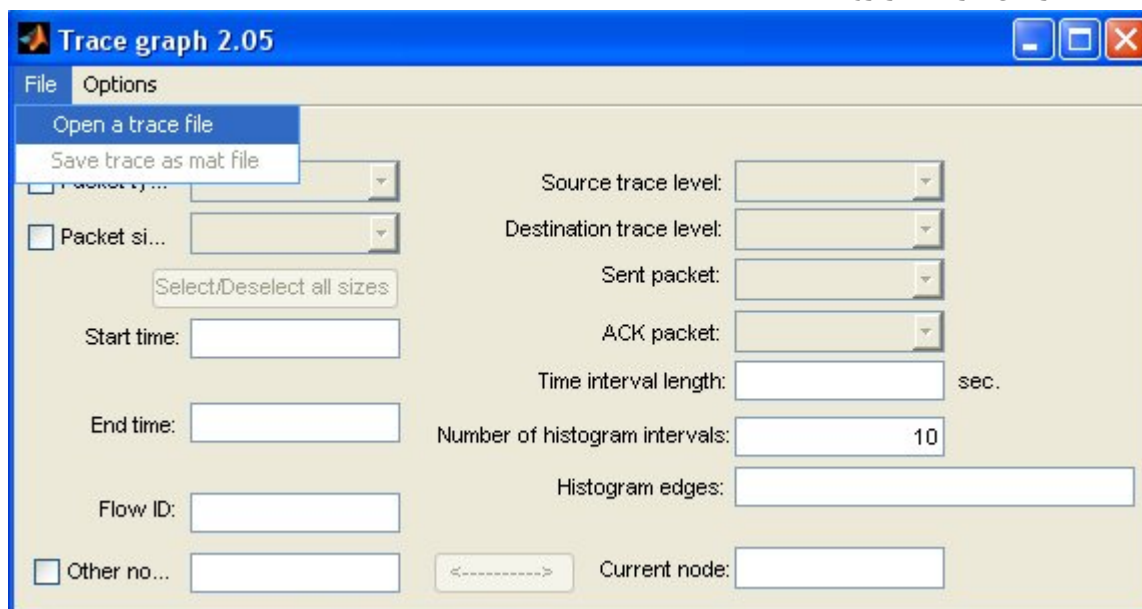
```


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---

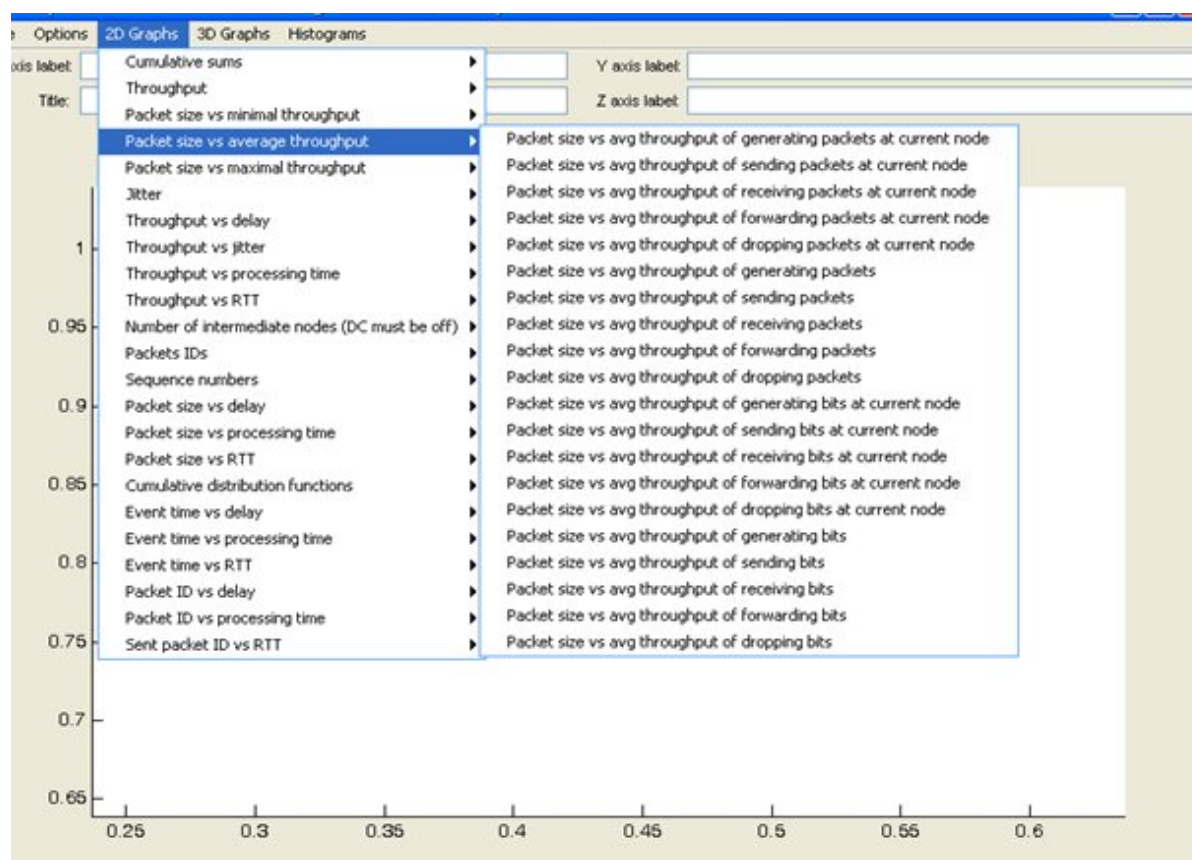


شکل ۳۸: محیط نرم افزار xgraph


بعد از این مرحله بر روی file->open کلیک کنید.




	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

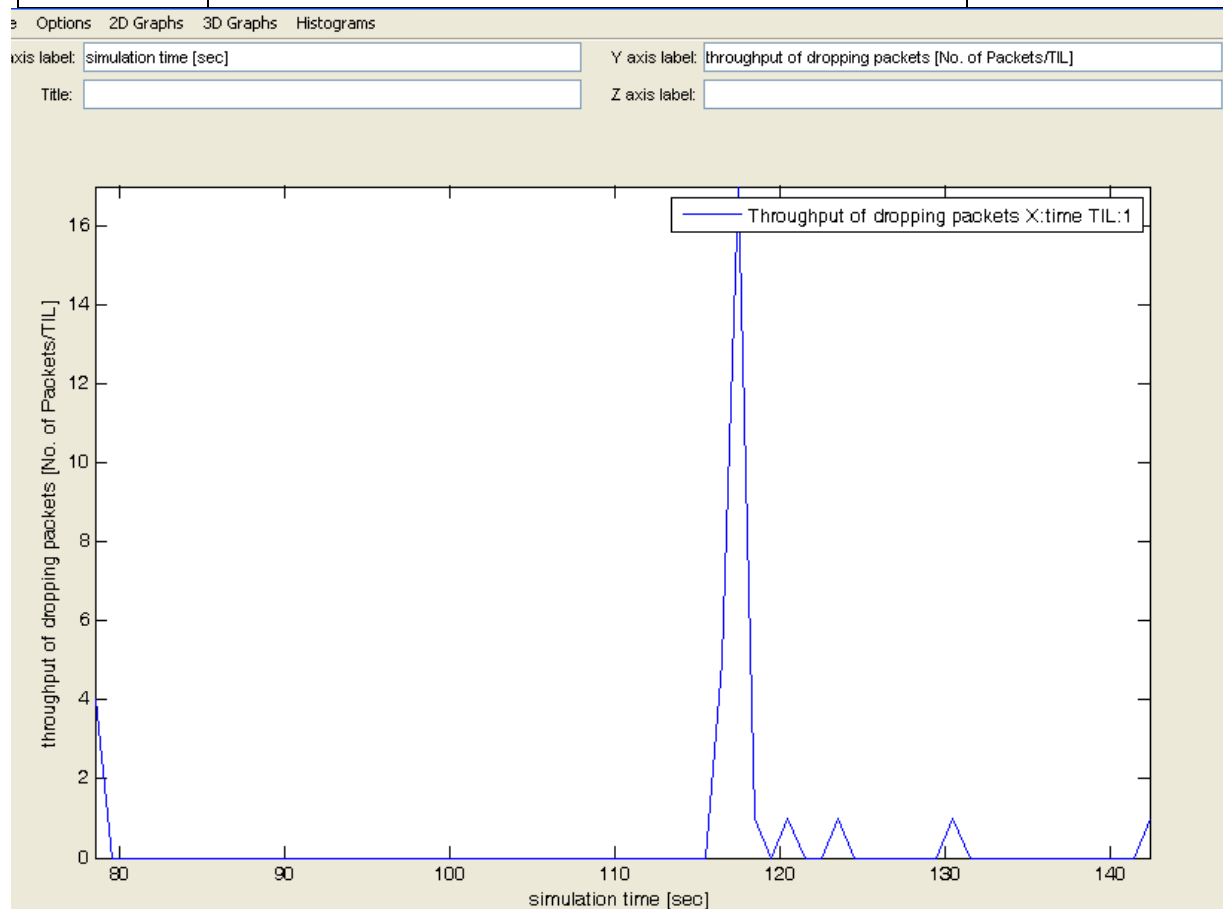


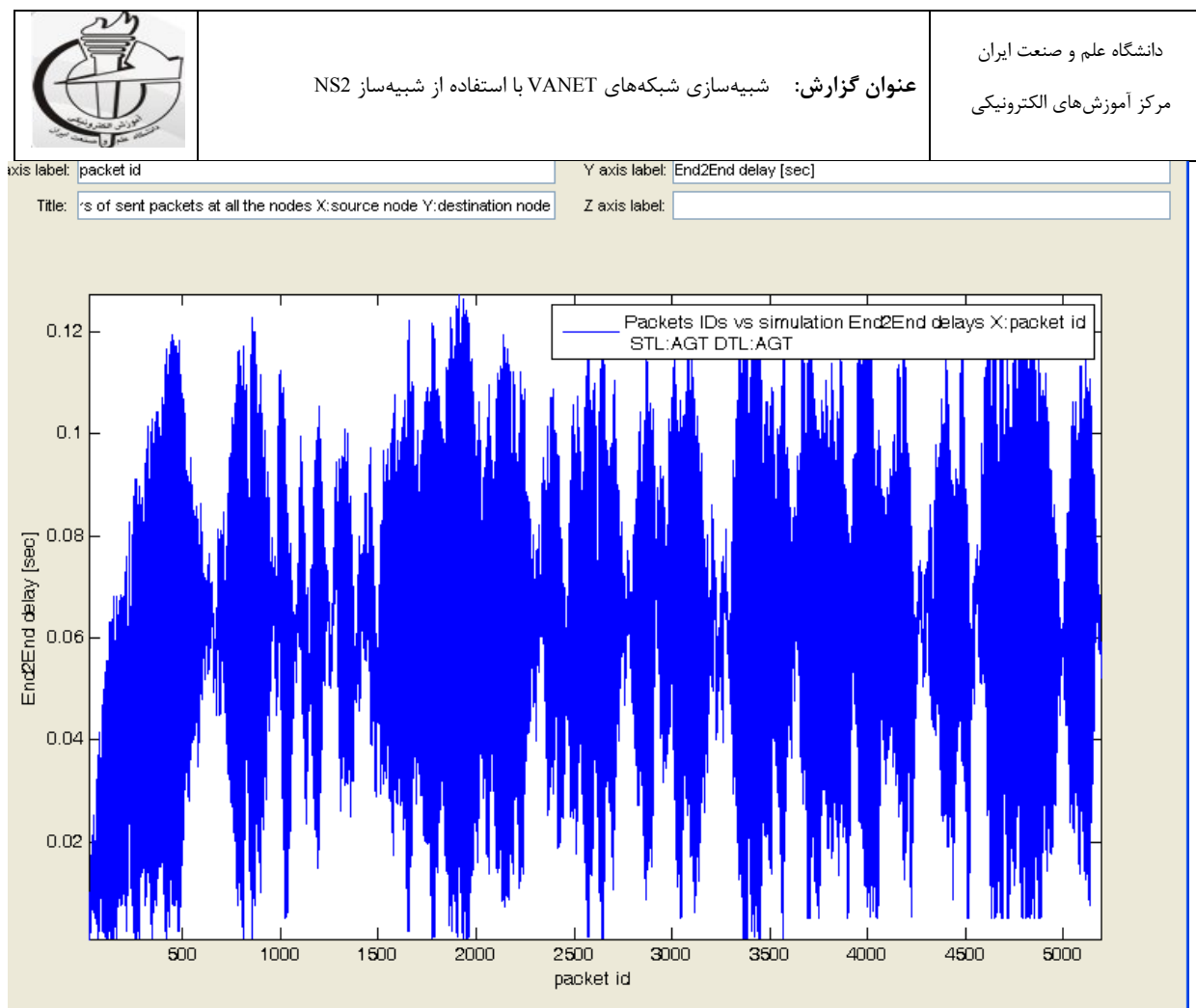
شکل ۳۹: باز کردن فایل Trace


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

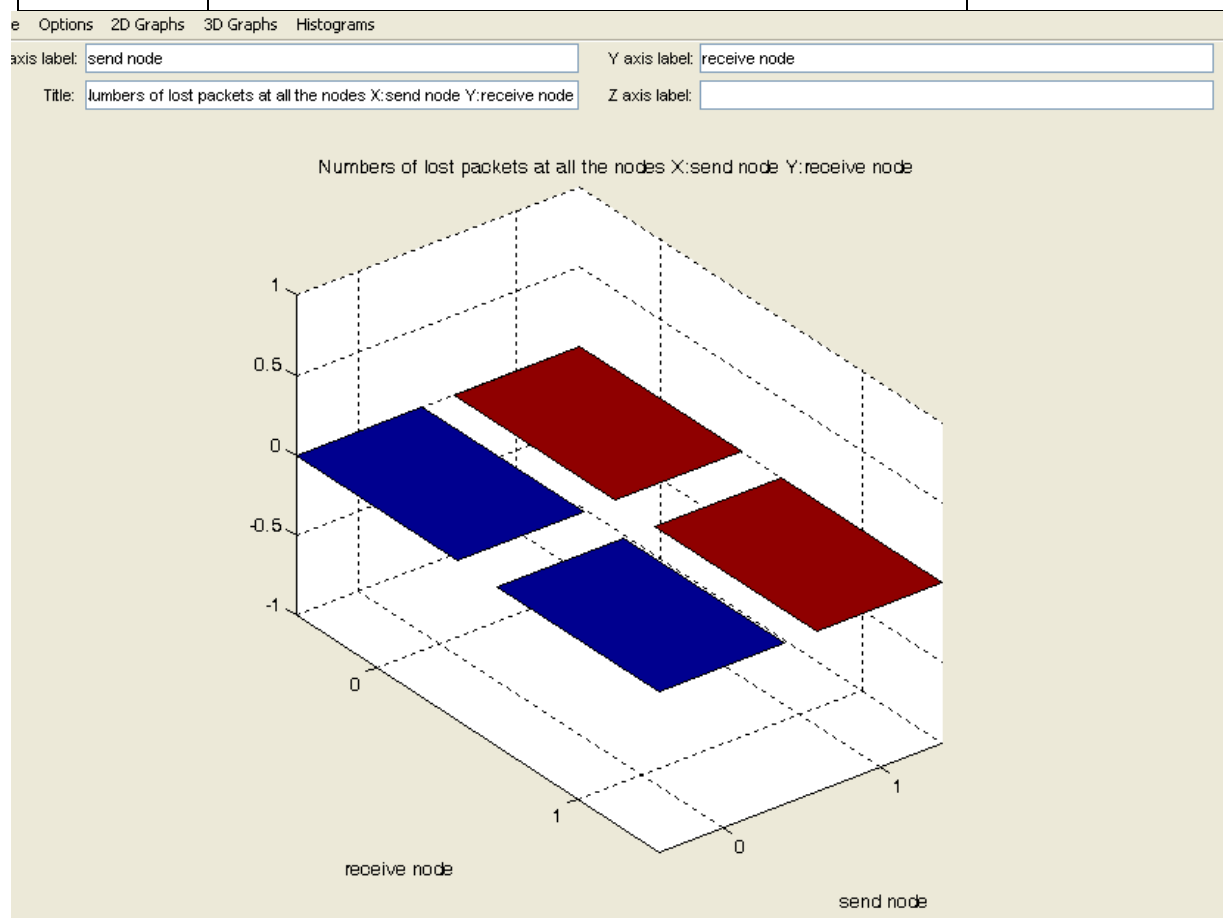
بعد از این مرحله پنجره بزرگتر فعال می شود که پارامترهای زیادی را در آن مشاهده میکنید که برای خروجی گرفتن می توانید استفاده کنید

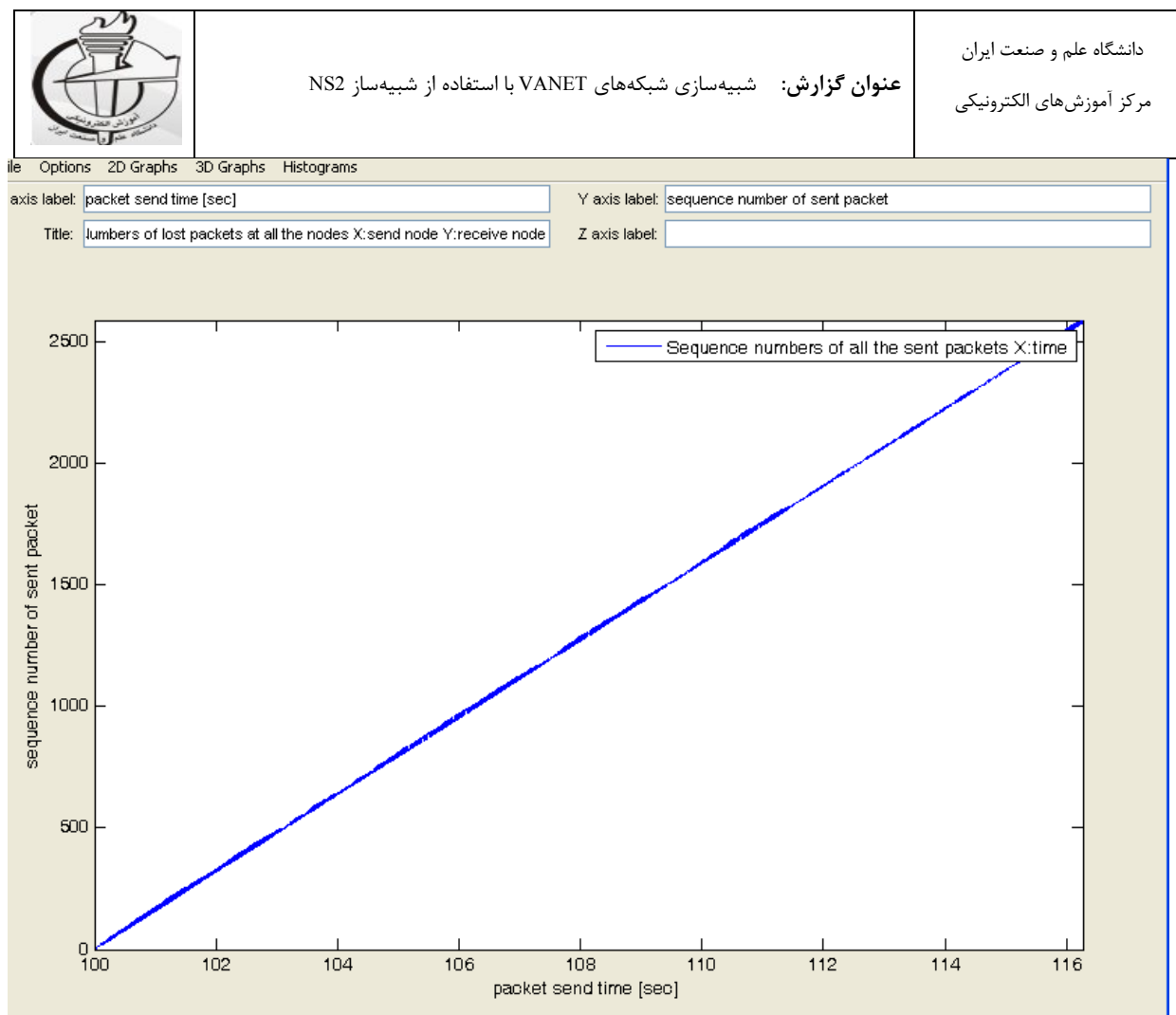
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---






	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---






	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

۵-۳- نمونه شبیه سازی شده ۲

در این بخش یک نمونه دیگر از برنامه tcl برای شبکه های vanet ارائه شده است این برنامه یک پروتکل broadcast را برای vanet شبیه سازی کرده است.

```
=====#
#
#Wireless simulation, a simulation of a broadcast protocol
#for vehicular ad hoc networks.
#
#Nathan Balon
#University of Michigan - Dearborn
#CIS 695
#
=====#
=====#
#Options
=====#

set opt(chan) Channel/WirelessChannel ;# channel type
set opt(prop) Propagation/TwoRayGround ;# radio-propagation model
#set opt(prop) Propagation/Shadowing
set opt(ant) Antenna/OmniAntenna ;# antenna type
set opt(ll) LL ;# link layer
set opt(ifq) Queue/DropTail/PriQueue ;# interface queue
set opt(ifqlen) 50 ;# max packet in ifq
set opt(netif) Phy/WirelessPhy ;# network interface type
set opt(mac) Mac/802_11 ;# mac type
set opt(rp) DumbAgent ;# routing protocol
set opt(nn) 50 ;# number of mobile nodes
set opt(pkt_size) 250 ;# size of broadcast message
set opt(cp) traffic.tcl ;# connection pattern
set opt(tr) trace.tr ;# trace file
set opt(nam_tr) "" ;# name trace file
set opt(seed) #...seed the random number generator
set opt(stop) 65.0 ;# time to end simulation
set opt(x) 1000 ;# x size for topology
set opt(y) 1000 ;# y size for topology
set opt(lm) ON ;# log movement
set opt(at) ON ;# agent trace
set opt(rt) OFF ;# routing trace
set opt(mact) ON ;# mac trace
set opt(movt) ON ;# movement trace
set opt(cw) 15
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

```
Mac/802_11 set CWMax_      1023
Mac/802_11 set SlotTime_   0.000013      ;# 20us
Mac/802_11 set SIFS_       0.000032      ;# 10us
Mac/802_11 set PreambleLength_ 32          ;# 144 bit
Mac/802_11 set PLCPHeaderLength_ 40        ;# 48 bits
Mac/802_11 set PLCPDataRate_ 6.0e6        ;# 1Mbps
Mac/802_11 set dataRate_   6.0e6
Mac/802_11 set basicRate_  6.0e6
```

```
Mac/802_11 set RTSThreshold_ 3000      ;# bytes
Mac/802_11 set ShortRetryLimit_ 7       ;# retransmissions
Mac/802_11 set LongRetryLimit_ 4        ;# retransmissions
```

```
Phy/WirelessPhy set CPTresh_ 10.0
Phy/WirelessPhy set CSTresh_ 2.5118864e-13 ;# -96 dBm
Phy/WirelessPhy set RXThresh_ 1.0e-12      ;# -90 dBm
Phy/WirelessPhy set bandwidth_ 6.0e6
Phy/WirelessPhy set Pt_ 0.0003754
Phy/WirelessPhy set freq_ 5.9e+9
Phy/WirelessPhy set L_ 1.0
```

```
#Propagation/Shadowing set pathlossExp_ 2.7
#Propagation/Shadowing set std_db_ 4.0
#Propagation/Shadowing set seed_ 0
#Propagation/Shadowing set dist0_ 1.0
```

```
#Unity gain, omni-directional antennas
#Set up the antennas to be centered in the node and 1.5 meters above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 4.0
Antenna/OmniAntenna set Gr_ 4.0
```

```
=====
```


```
set BROADCAST_ADDR -1      ;# broadcast address
```

```
set MESSAGE_PORT 42        ;# periodic message port
set WARNING_PORT 33        ;# warning message
set EMER_PORT 21           ;# port to listen for emergency messages
```

```
Class Agent/MessagePassing/PeriodicBroadcast -superclass Agent/MessagePassing
```

```
Agent/MessagePassing/PeriodicBroadcast instproc recv {source sport size data} {
```

```
# This empty function is needed so receive works.
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

Agent/MessagePassing/PeriodicBroadcast instproc send_message {} {}
$ self instvar node_
global ns_ MESSAGE_PORT BROADCAST_ADDR opt

#puts "[ $node_ node-addr] sending message"
# send the broadcast message
$ self sendto $opt(pkt_size) 0 $BROADCAST_ADDR $MESSAGE_PORT
}

#perform clean up at the end of the program
proc finish {} {}
    global ns_ tracefd namtracefd opt
$ ns_ flush-trace
close $tracefd
if {$opt(nam_tr)"" != ""} {}
    close $namtracefd
}
$ ns_ halt
exit 0
}


#set the options from the command line arguments
proc getopt {argc argv} {
    global opt
    lappend optlist cp nn seed sc stop tr x y bc_size nam_tr cw

    for {set i 0} {$i < $argc} {incr i} {
        set arg [lindex $argv $i]
        if {[string range $arg 0 0] != "-"} continue
        set name [string range $arg 1 end]
        set opt($name) [lindex $argv [expr $i+1]]
    }
}

#log the movement of a node every 0.1 seconds
proc log-movement {} {}
    global logtimer ns_ ns

    set ns $ns_
    source ../tcl/mobility/timer.tcl
    Class LogTimer -superclass Timer
    LogTimer instproc timeout {} {}
    global opt node;_

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

for {set i 0} {$i < $opt(nn)} {incr i} {
    $ node_($i) log-movement
}
$ self sched 0.1
}

set logtimer [new LogTimer]
$ logtimer sched 0.1
}
#get command line arguments
getopt $argc $argv

Mac/802_11 set CWMin_ $opt(cw (

puts "x: $opt(x), y: $opt(y)"
puts "CW: $opt(cw)"

if {$opt(seed) > 0} {
    puts "Seeding Random number generator with $opt(seed)\n"
    ns-random $opt(seed)
}

#create a new simulator
set ns_ [new Simulator]

#set up the traces
set tracefd [open $opt(tr) w]
$ns_ trace-all $tracefd


#set the topology
set topo [new Topography]
$topo load_flatgrid $opt(x) $opt(y)

set god_ [ create-god $opt(nn)]
set chan_ [new Channel/WirelessChannel]

#Set up the nam trace if desired.
if {$opt(nam_tr)"" !=!} {
    set namtracefd [open $opt(nam_tr) w]
    $ ns_ namtrace-all-wireless $namtracefd $opt(x) $opt(y)
}

#configure the nodes of the simulation
$ns_ node-config -adhocRouting $opt(rp)\

```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---

```

- llType $opt(ll)\
- macType $opt(mac)\
- ifqType $opt(ifq)\
- ifqLen $opt(ifqlen)\
- antType $opt(ant)\
- propInstance [new $opt(prop)]\
- phyType $opt(netif)\
- topoInstance $topo\
- channel $chan\ _
- agentTrace $opt(at)\
- routerTrace $opt(rt)\
- macTrace $opt(mact)\
- movementTrace $opt(movt)

```

```
set y 0
```

```
set x 0
```

```
#Create the mobile nodes for the simulation.
```

```

for {set i 0} {$i < $opt(nn)} {incr i} {
    set node_($i) [$ns_ node[
$   ns_ initial_node_pos $node_($i) 10
}

```

```
#Source the mobility scenario file.
```

```
source $opt(sc)
```

```
#Attach a new Agent/MessagePassing/PeriodicBroadcast to each node on port $MESSAGE_PORT
```

```

for {set i 0} {$i < $opt(nn)} {incr i} {
    set bc_agent($i) [new Agent/MessagePassing/PeriodicBroadcast[
$   node_($i) attach $bc_agent($i) $MESSAGE_PORT
}

```

```
#log movement
```

```
if { $opt(lm) == "ON " }
```

```
puts "Logging movement"...
```


```
log-movement
```

```
}
```

```
#Source the file that contains the broadcast traffic.
```

```
source $opt(cp)
```

```
$ns_ at $opt(stop) "finish"
```


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	---	---


```
#Add to the trace simulation parameters .
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp $opt(rp)"
#puts $tracefd "M 0.0 cp $opt(cp) seed $opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"
puts "Starting Simulation"...
$ns_run
```

۵-۴- نمونه شبیه سازی شده ۳

همانطور که در قسمت قبل اشاره شد. اضافه کردن موبیلتی به گره ها در ns2 بسار پیچیده است و نمی شود به سادگی آن را در محیط پیاده سازی کرد. برای این بهتر است از شبیه ساز ns2 به همراه شبیه ساز SUMO استفاده کرد. نرم افزار SUMO برای شبیه سازی مدل های حرکتی سیار شهری است. این نرم افزار می تواند جنبه های مختلف یک شبکه را با جزییات پیاده سازی کند و نقشه را به مدل های حرکتی تبدیل کرد یا اطلاعات را از پایگاه های داده ای سیستم اطلاعات جغرافیایی خوانده و در ns2 از آنها استفاده کرد. اگر چه sumo نرم افزار قدرتمندی با قابلیت های بسار زیاد در ایجاد مدل های حرکتی شبکه ای سیار شهری است اما فاقد محیط گرافیکی برای تولید چنین محیط هایی است. برای کار با آن باید به صورت دستی فایل های حاوی مشخصات خودروها و جاده ها و فایل های پیکربندی را ایجاد کرده و معمولا به صورت xml نوشته و در یک فایل sumi, cfg ذخیره می شود و با دستورات متنی آنها را به هم لینک داده یا فایل های مورد نیاز تولید کرد.

برای دریافت نرم افزار به لینک زیر رفته و شبیه ساز sumo را دانلود کنید.

<http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Downloads>



عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2

دانشگاه علم و صنعت ایران


مرکز آموزش های الکترونیکی

page

discussion

view source

history



navigation

- Main Page
- Community portal
- Current events
- Recent changes
- Random page
- Help

search

Go

Search

toolbox

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link

Downloads

Here, you will find SUMO both as sources and as compiled binaries. As the program is still under development and is extended continuously, we advice you to use the latest sources from our [Subversion Repository](#) . Normally, they should compile and be newer than the releases are.

Please contact us if you have any problems. If you want to report a bug, please use the [Sourceforge BugTracker](#) . For further information about the changes between releases see the [ChangeLog](#).

SUMO is licensed under the [GPL](#) .

Nightly Builds

The code within the SVN is compiled each night. The following resulting packages can be obtained:

- <http://sumo.sourceforge.net/daily/sumo-msvc8Win32-bin.zip> (windows, 32bit)
- <http://sumo.sourceforge.net/daily/sumo-msvc8x64-bin.zip> (windows, 64bit)

SUMO - Latest Release (Version 0.11.0.1)

Release date: 30.07.2009

- The latest MS Windows binaries.

Contains the binaries, all dlls needed, the examples, tools, and documentation in pdf format.

Download as: [sumo-winbin-0.11.0.1.zip](#), ~8.29MB
- Linux binaries are created by the [opensUSE build service](#)

The repositories also contain the libraries (like proj and gdal) if they are not part of the distribution. At the moment there is no documentation included in the packages.
 - [opensUSE 10.3 repository](#)
 - [opensUSE 11.0 repository](#)
 - [opensUSE 11.1 repository](#)
 - [Fedora 9 repository](#)
- The latest source-tarball.

Includes sources, examples, Linux-Makefiles and solution file for MSVC++8.0. Does not contain tests.

Download as:
 - [sumo-src-0.11.0.1.tar.gz](#), ~4.02MB
 - [sumo-src-0.11.0.1.zip](#), ~5.13MB
- The latest documentation-tarball. Includes generated pdfs.

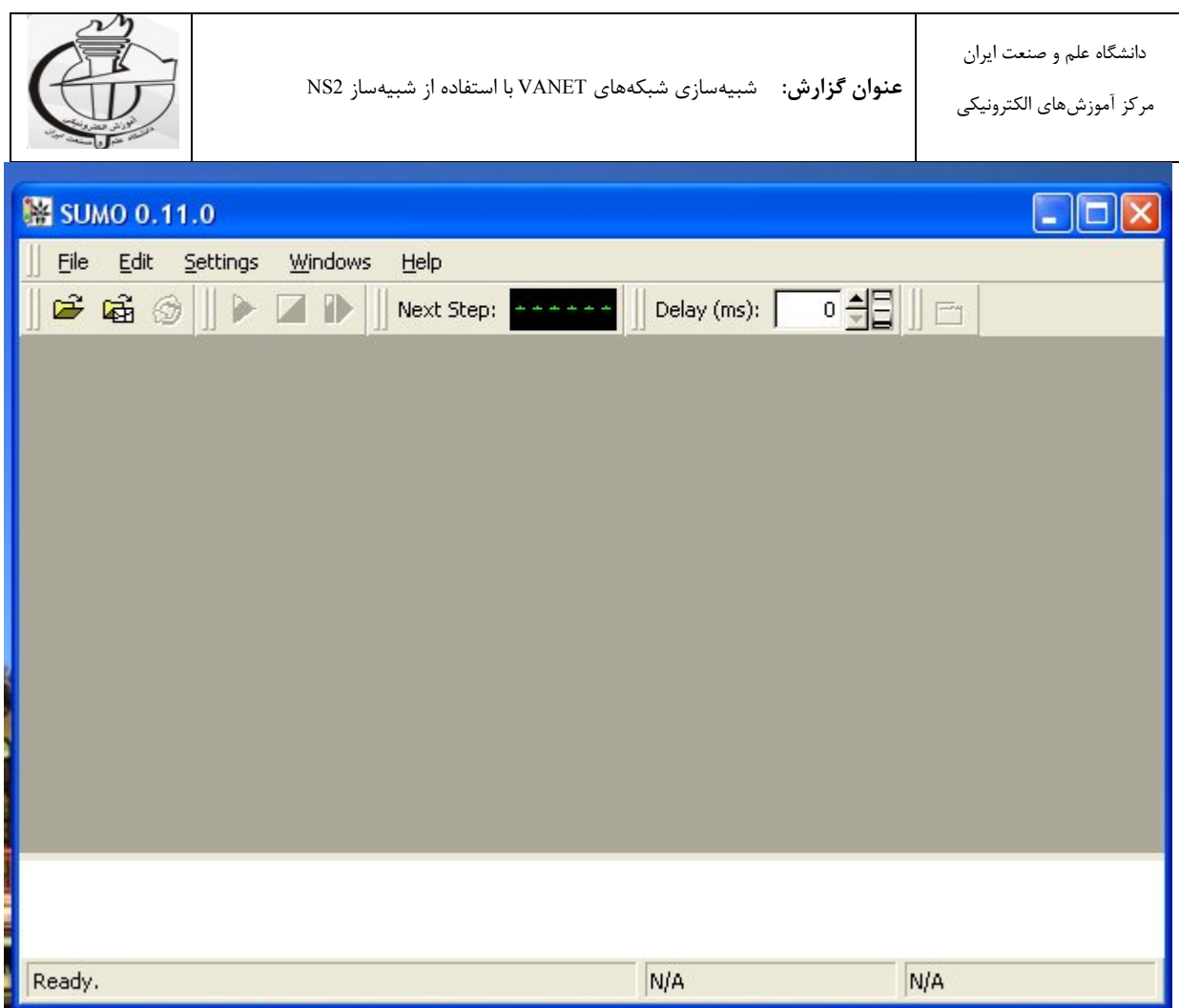
Download as: [sumo-doc-0.11.0.1.tar.gz](#), ~1.06MB
- The latest tests-tarball. These tests run with [TextTest](#) .

Download as: [sumo-tests-0.11.0.1.tar.gz](#), ~6.89MB

SUMO - older releases

شکل ۴۰: صفحه دانلود نرم افزار sumo


لازم به ذکر است مدل جدید دارای محیط گرافیکی می باشد.




شکل ۴۱: محیط نرم افزار sumo

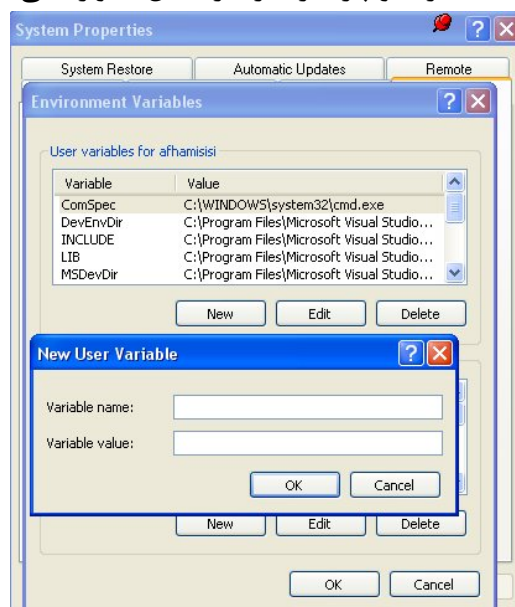
نصب sumo در محیط ویندوز بسیار ساده است شما می توانید فایل های باینری را از ادرس آمده در بخش قبل که مربوط به سیستم عامل لینوکس می باشد دانلود کنید. باید قبل از نصب نرم افزار باید نرم افزار Microsoft Visual C++ 2005 SP1 را نصب کنید. بعد از نصب نرم افزار در ENVIROMENT ویندوز پارامترهای زیر را باید حتما وارد کنید.

```
Path=
C:\Program Files\Microsoft Visual Studio 8\Common7\IDE;C:\Program Files\Microsoft Visual Studio
8\VC\BIN;C:\Program Files\Microsoft Visual Studio 8\Common7\Tools;C:\Program Files\Microsoft Visual
Studio 8\Common7\Tools\bin;C:\Program Files\Microsoft Visual Studio 8\VC\PlatformSDK\bin;C:\Program
Files\Microsoft Visual Studio
8\SDK\v2.0\bin;C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;C:\Program Files\Microsoft Visual
Studio 8\VC\VCpackages;C:\Program Files\Microsoft Visual Studio
8\SDK\v2.0\Bin;C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;C:\Program Files\Microsoft Visual
Studio 8\VC\bin;C:\Program Files\Microsoft Visual Studio 8\Common7\IDE;C:\Program Files\Microsoft Visual
Studio 8\VC\vcpackages
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre> INCLUDE= C:\Program Files\Microsoft Visual Studio 8\VC\ATLMFC\INCLUDE;C:\Program Files\Microsoft Visual Studio 8\VC\INCLUDE;C:\Program Files\Microsoft Visual Studio 8\VC\PlatformSDK\include;\Program Files\Microsoft Visual Studio 8\SDK\v2.0\include LIB= C:\Program Files\Microsoft Visual Studio 8\VC\ATLMFC\LIB;C:\Program Files\Microsoft Visual Studio 8\VC\LIB;C:\Program Files\Microsoft Visual Studio 8\VC\PlatformSDK\lib;\Program Files\Microsoft Visual Studio 8\SDK\v2.0\lib LIBPATH= C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;C:\Program Files\Microsoft Visual Studio 8\VC\ATLMFC\LIB NetSamplePath= C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0 DevEnvDir= C:\Program Files\Microsoft Visual Studio 8\Common7\IDE FrameworkDir= C:\WINDOWS\Microsoft.NET\Framework FrameworkSDKDir= C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0 FrameworkVersion= v2.0.50727 VCBUILD_DEFAULT_CFG= Debug^ Win32 VCBUILD_DEFAULT_OPTIONS= /useenv VCINSTALLDIR= C:\Program Files\Microsoft Visual Studio 8\VC </pre>		

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<p>VSINSTALLDIR= C:\Program Files\Microsoft Visual Studio 8</p>		

پس از نصب نرم افزار Microsoft Visual C++ و تغییر پارامترهای بالا در ویندوز (برای بر روی my computer کلیک راست کرده و گزینه properties را انتخاب می کنید و از انجا به منوی advanced رفته و Environment Variables را انتخاب کرده و پارامترها را در شکل () وارد می کنید).




نرم افزار sumo داندلود شده را از حالت فشرده خارج می شود. به همین سادگی نرم افزار نصب می شود. در صورت خطا دادن در این قسمت حتما باید به نصب Microsoft Visual C++ و مراحل وارد کردن مشخصات نصب خود توجه کنید.

در این بخش با مالی بسیار ساده با اصول کار شبیه ساز sumo آشنا می شویم. در این مثال که ساده ترین حالت ممکن را ایجاد می کند. یک خودرو را در حال رانندگی نشان می دهد. برای نوشتن برنامه از برنامه نویسی استفاده می شود. در شبیه ساز sumo، شبکه خیابانها از گره ها^۱ و لبه ها^۲ تشکیل شده است. لبه ها، گره ها را به هم متصل می ند. بنابراین در یک شبکه که می خواهیم دو خیابان را به متصل شود به سه گره و دو لبه نیاز داریم.

گره ها حاوی دو مشخصه طول- عرض جغرافیایی (فاصله تا مبدا به متر) و شناسه (id) برای دسترسی های بعدی نیازمند است.

¹ node

² Edge

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

<pre><nodes> <node id="1" x="-500.0" y="0.0" /> <node id="2" x="+500.0" y="0.0" /> <node id="3" x="+501.0" y="0.0" /> </nodes></pre>	برنامه xml برای پیکربندی شبکه
--	-------------------------------

برای اینکار می توانید از یک ویرایشگر همانند notepad یا PSeditor استفاده کنید. برای دانلود نرم افزار PSeditor می توانید از لینک زیر استفاده کنید.

<http://www.softpedia.com/get/Office-tools/PDF/PSEDITOR.shtml>

برنامه را به اسم hello.node.xml ذخیره کنید که node.xml پسوند پیش فرض برای فایل حاوی تنظیمات گره در sumo می باشد.


در مرحله بعد، گره ها را توسط لبه ها به هم متصل می کنیم. این کار بسیار ساده می باشد از دو مشخصه fromnode و tonode استفاده می شود که در آن fromnode گره ابتدای لبه و tonode گره پایانی گره می باشد.

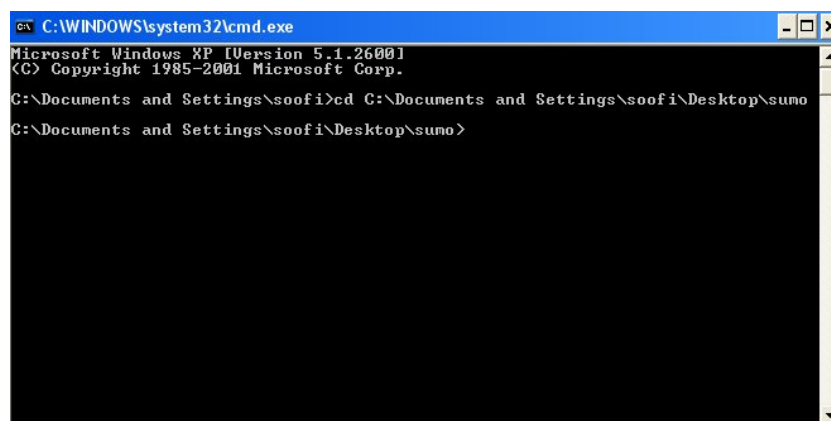
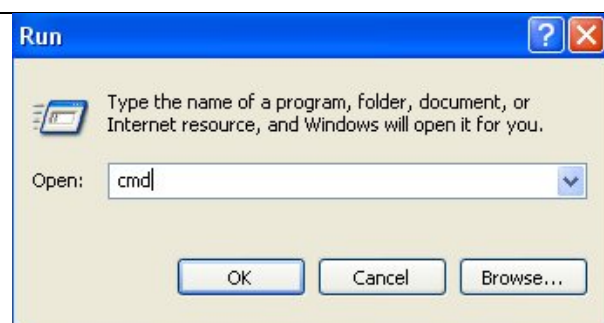
<pre><edges> <edge fromnode="1" id="1to2" tonode="2" /> <edge fromnode="2" id="out" tonode="3" /> </edges></pre>	ارتباط میان لبه و گره
--	-----------------------

مشخصه id برای دسترسی های بعدی می باشد. داده ها را در فایل hello.edge.xml ذخیره کنید. بعد از ایجاد گره ها و لبه ها، می توان از یکی از ابزار sumo به نام netconvert برای ایجاد کردن شبکه استفاده کنید.

بدین منظور ابتدا به محیط command prompt رفته و به مسیر نصب فایل تغییر مسیر می دهید. (طبق


شکل (۱))

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	--	---



پس از این مرحله دستور زیر را می نویسید.

```
netconvert -xml-node-files=hello.nod.xml -xml-edge-files=hello.edg.xml -output
file= hello.net.xml
```

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

پس از این دستور فایل hello.net.xml ایجاد می کند. بعد از این مرحله مسیرها ایجاد می شود. در sumo، خودروها دارای انواع مختلف هستند. زیرا خصوصیات اصلی آنها همانند طول، شتاب، حداکثر سرعت و... متفاوت هستند. علاوه بر این خودرو پارامتر به نام sigma نیاز دارد که یک رفتار تصادفی را با توجه به مدل خودرو تولید می شود. مقداردهی صفر به این مشخصه باعث تولید یک خودرو قطعی می شود. اکنون یک مسیر برای خودرو ایجاد می شود که دو لبه را در بر می گیرد. علت اینکه از دو لبه استفاده می شود این است که یک خودرو هنگامی که به یک لبه می رسد ناپدید می شود.

```
<routes>
<vtype accel="1.0" 25dged="5.0" id="Car" length="2.0" maxspeed="100.0"
sigma="0.0" />
<route id="route0">1to2 out</route>
<vehicle depart="1" id="veh0" route="route0" type="Car" />
</routes>
```

کدها در فایل hello.ru.xml ذخیره می شود.


۵-۴-۱- پیکربندی

در این مرحله تمامی فایل ها را در یک فایل پیکربندی با هم پیوند زده می شود.

```
<configuration>
<files>
<net-file>hello.net.xml</net-file>
<route-files>hello.rou.xml</route-files>
</files>
<simulation>
<begin>0</begin>
<end>10000</end>
</simulation>
</configuration>
```

پس از این مرحله در فایل hello.sumo.cfg شبیه سازی را از طریق محیط command promپ یعنی در محیط متنی اجرا کرد.

```
sumo -c hello.sumo.cfg
```


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

یا با دستور زیر در محیط گرافیکی sumo اجرا کرد.

Guisim -c hello.sumo.cfg

بعد از این مرحله باید نرم افزار move را دانلود کنید برای دانلود نرم افزار move از لینک زیر می توانید استفاده کنید.

<http://lens1.csie.ncku.edu.tw/MOVE/index.htm>

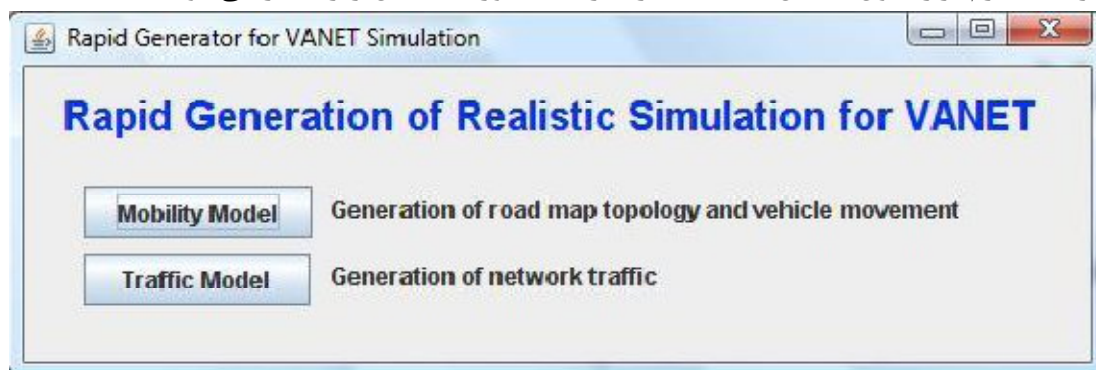
برای اجرا این نرم افزار باید نرم افزارهای مربوط به جاوا باید حتما دانلود شود. از انجا کشور ما تحریم می باشد نمی توان آن را از سایت جاوا مستقیما دانلود کرد. سایت www.p30download.com از قسمت نرم افزارهای برنامه نویسی دانلود کرد. Move با نرم افزار جاوا نوشته شده است. این بدان معنا است که بر روی سیستم عاملهای مختلف همانند لینوکس و ویندوز قابل پیاده سازی است. برای کامپایل کردن move ابتدا باید همانند نرم افزار ++C Visual C++ microsoft طابید مسیرهای آن وارد سیستم شود. پس از آن در محیط متنی دستور زیر نوشته شود.

java -jar MOVE.jar

برای اجرای آن دستور زیر را وارد کنید.

java vanetsim


اگر نصب نرم افزار بدون مشکل باشد. بعد از اجرای دستور بالا شکل زیر ظاهر می شود.



نرم افزار move واسط میان نرم افزار میان ns2 و sumo است. در ادامه به توضیح آن اشاره می شود.


۵-۴-۲- مراحل شبیه سازی

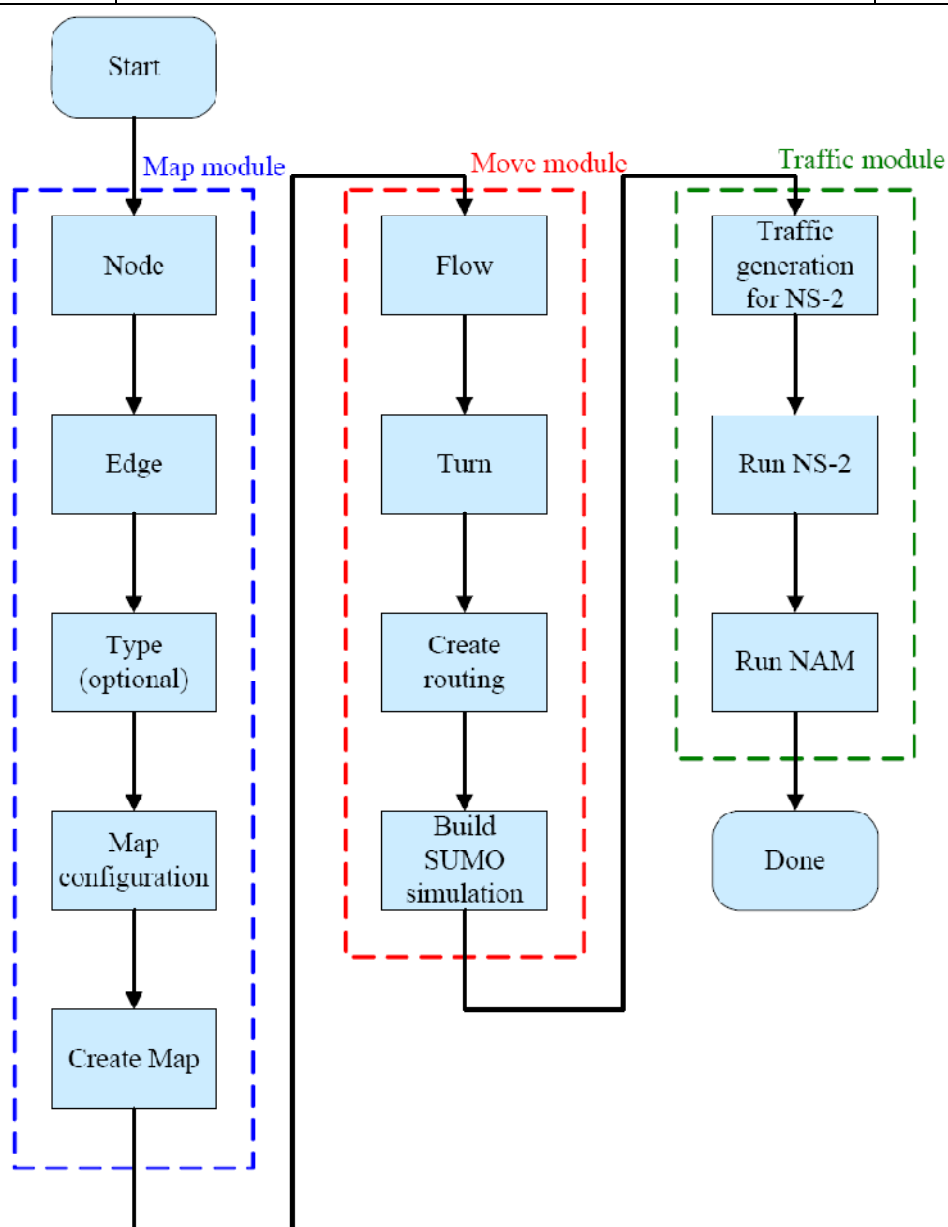
برای شبیه سازی VANET ، ابتدا باید مدل حرکتی را با استفاده از SUMO تولید می کنیم و سپس این کار را با واسط گرافیکی MOVE انجام داده می شود. شکل زیر مراحل انجام کار را با استفاده از ماژول های مختلف MOVE انجام می شود.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
---	---	---



شکل بعد مازولهای مختلف نرم افزار MOVE را نشان می دهد.


	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---

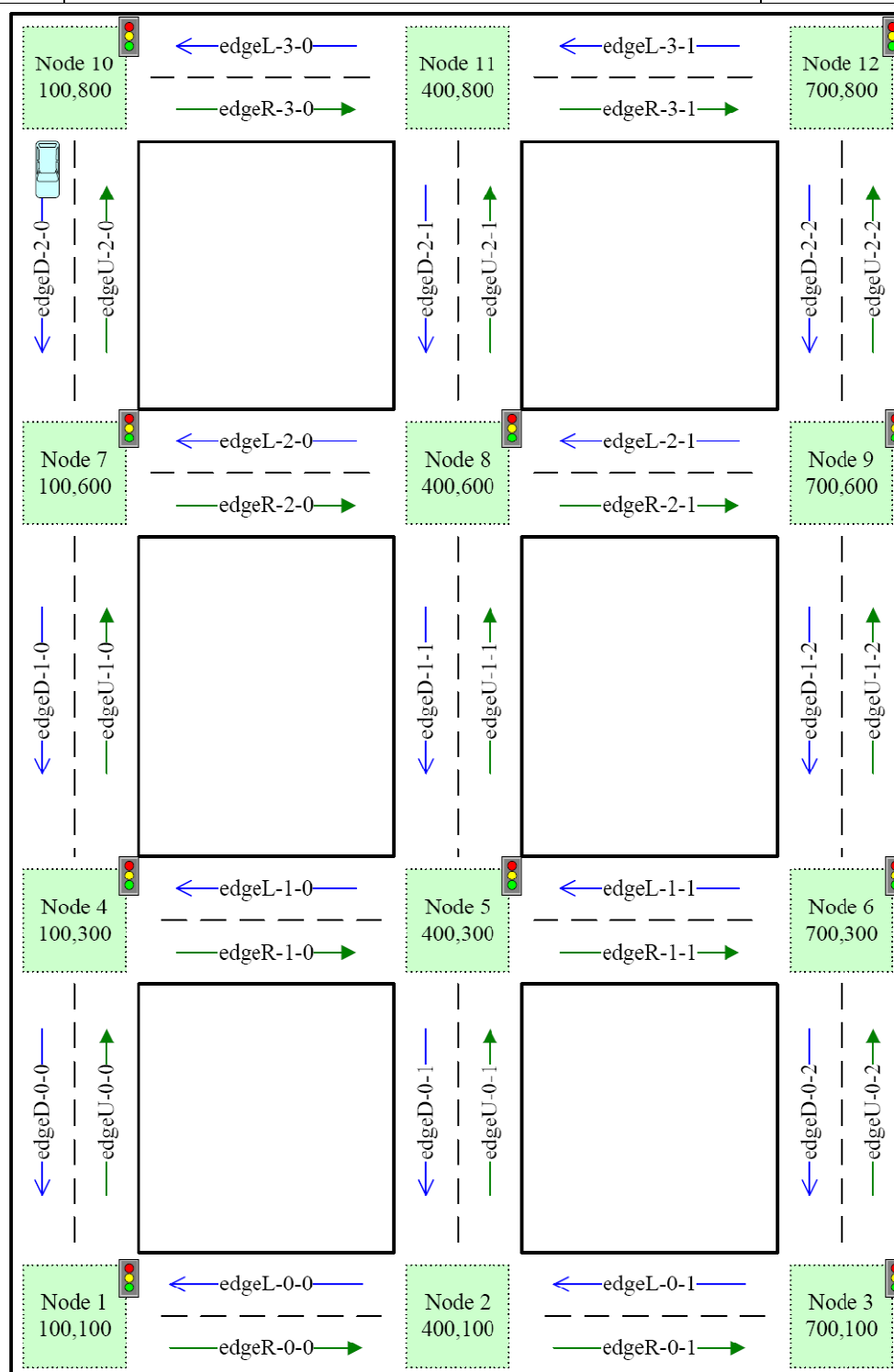


شکل ۴۲: ارتباط نرم افزار SUMO و NS2 و MOVE

۳-۴-۵- پیاده سازی یک مثال

فرض کنید که می خواهیم مدل زیر را در شبکه پیاده سازی کنیم.

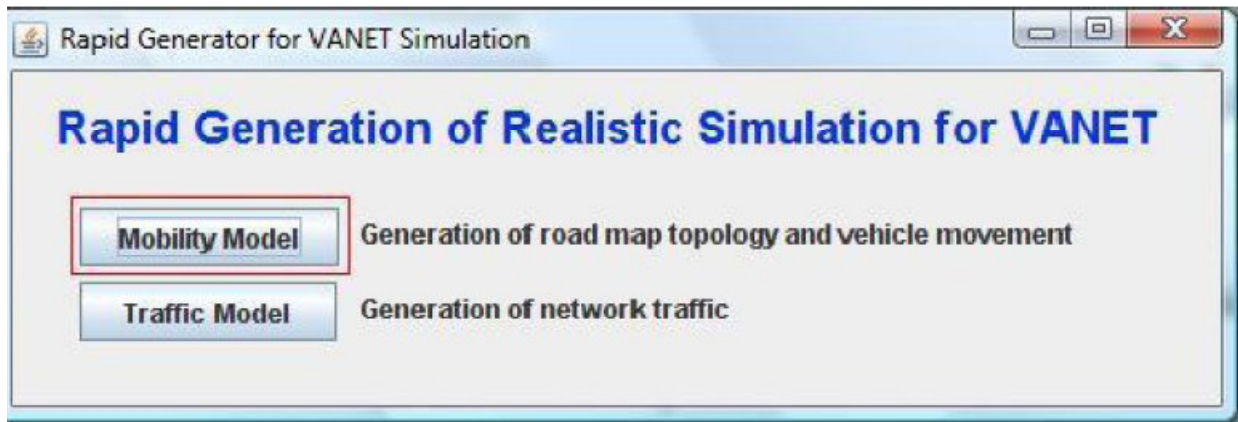
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
---	---	---




همانطور که در شکل ۴۲ نشان داده شده است. MOVE از دو بخش Mobility Model و Traffic Model تشکیل شده است. Mobility Model برای ایجاد مدل حرکتی مورد نیاز بکار می رود که این کار را

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

برای ارتباط با نرم افزار sumo انجام می شود. و منوی دوم Traffic Model مدل ترافیکی برای شبکه های نرم افزاری NS2 تولید و اجرا می کند.




در ابتدا Mobility Model انتخاب می شود. انتخاب Mobility Model برای کار با نرم افزار sumo استفاده می شود. ابتدا باید مسیر SUMO را برای آن مشخص شود. از منوی Set Environment سیستم عامل مورد نظر خود را انتخاب کنید.

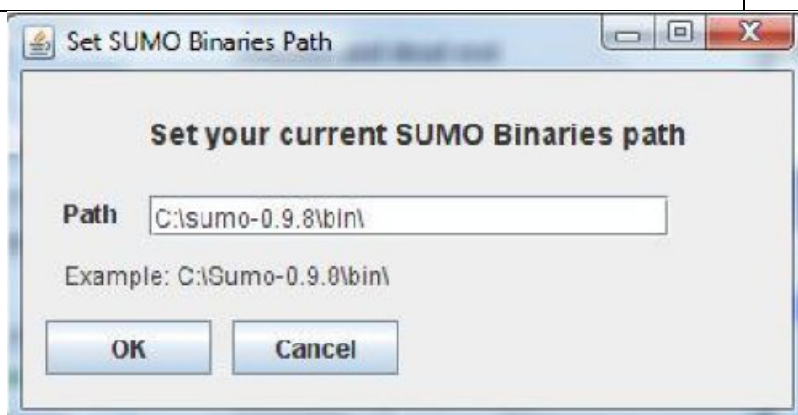
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---



شکل ۴۳: انتخاب سیستم عامل سیستم

توجه کنید اگر از سیستم ویندوز استفاده می کنید باید حتما مسیر نرم افزار sumo را باید انتخاب کنید که در شکل ۴۴ نشان داده شده است.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

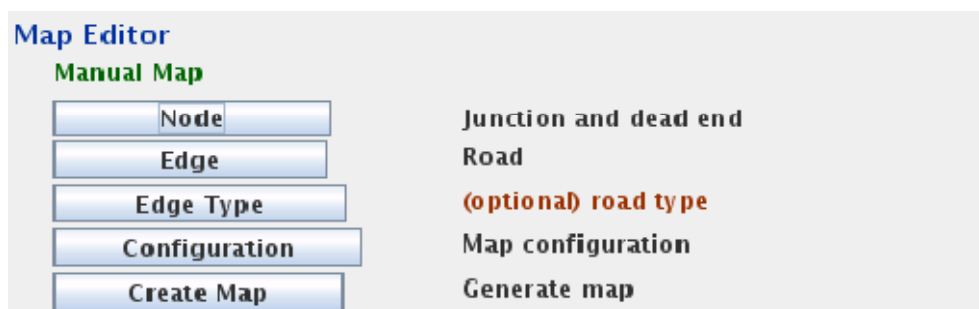


شکل ۴۴: مشخص کردن مسیر نرم افزار sumo در محیط نرم افزار move

در محیط لینوکس نیازی به وارد کردن مسیر ندارید. بعد از این مرحله شبیه ساز شما آماده است و می توانید نقشه شبکه را بکشید. در ادامه به شبیه سازی نرم افزار توسط شبیه ساز move پرداخته می شود.

۵-۴-۴- توليد نقشه

۵-۴-۴-۱- ایجاد نقشه به صورت دستی

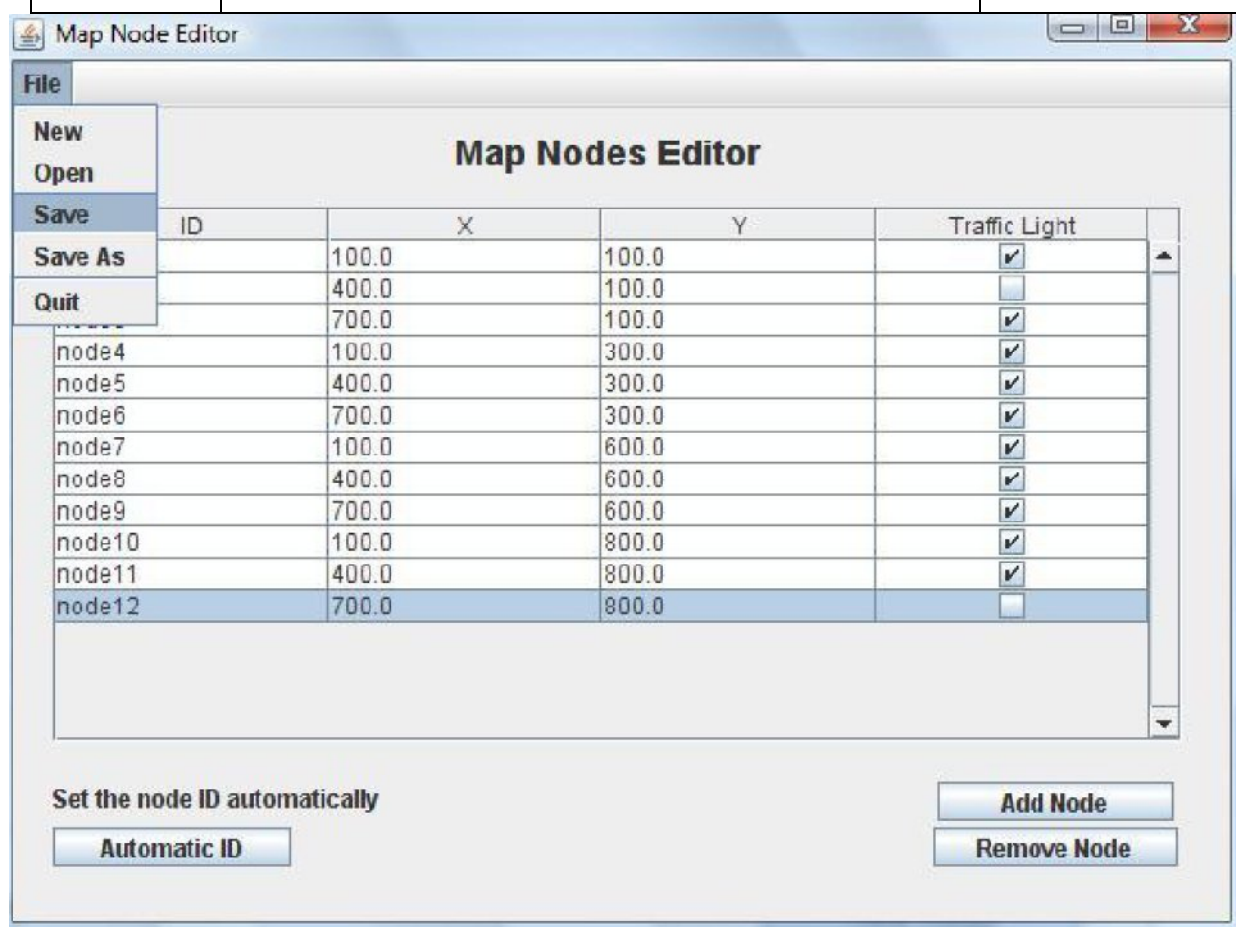


شکل ۴۵: ایجاد نقشه به صورت دستی

ابتدا بهتر است از منوی اصلی Node انتخاب شود. در این بخش می توان گره ها را به صورت دستی ایجاد کرد. بعد از انتخاب Node شکل ۴۵ را مشاهده می کنید.

دانشگاه علم و صنعت ایران
مرکز آموزش های الکترونیکی


عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2



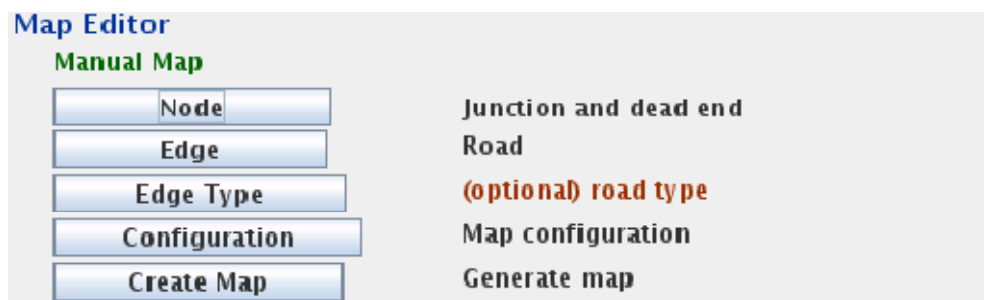
شکل ۴۶: وارد کردن گره ها به صورت دستی

همانطور که در بخش قبل توضیح داده شده است. در نرم افزار SUMO هر گره دارای مختصات و شناسه است که در شکل () نشان داده شده است. گزینه Traffic Light نیز مشخص می کند که خودروها چه عکس العملی در پیچ ها داشته باشند. بعد از وارد کردن گره ها باید آن را در فایل با پسوند node.xml ذخیره کنند. (به عنوان مثال ex_Node.node.xml) محتویات فایل حاوی گره ها در مثال ما در ادامه آمده است.

```
<nodes>
<node id="node1" x="100.0" y="100.0" type="traffic_light" />
<node id="node2" x="400.0" y="100.0" type="priority" />
<node id="node3" x="700.0" y="100.0" type="traffic_light" />
<node id="node4" x="100.0" y="300.0" type="traffic_light" />
<node id="node5" x="400.0" y="300.0" type="traffic_light" />
<node id="node6" x="700.0" y="300.0" type="traffic_light" />
<node id="node7" x="100.0" y="600.0" type="traffic_light" />
```

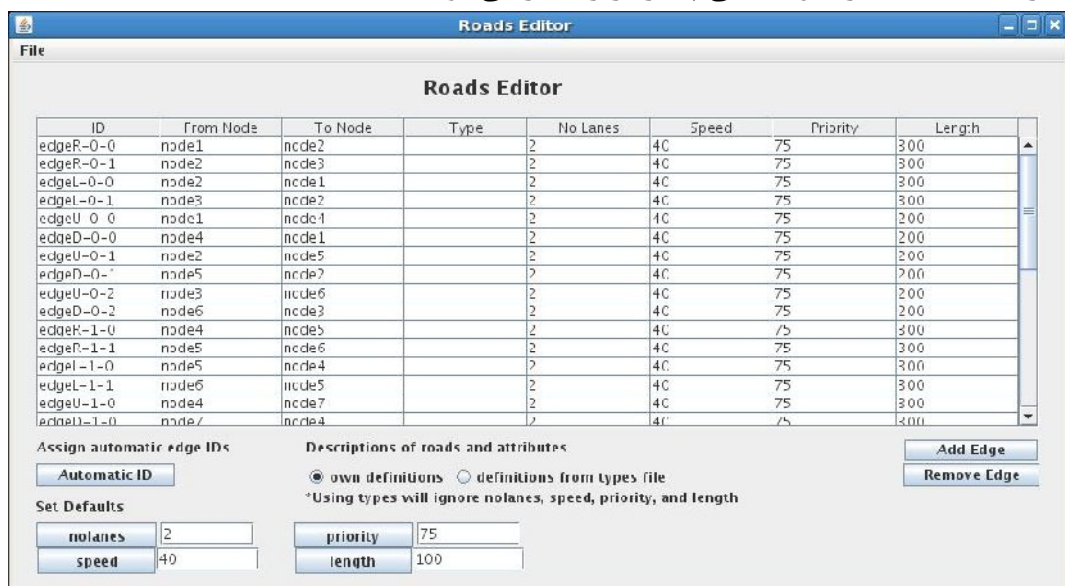

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre><node id="node8" x="400.0" y="600.0" type="traffic_light" /> <node id="node9" x="700.0" y="600.0" type="traffic_light" /> <node id="node10" x="100.0" y="800.0" type="traffic_light" /> <node id="node11" x="400.0" y="800.0" type="traffic_light" /> <node id="node12" x="700.0" y="800.0" type="priority" /> </nodes></pre>		
<p>ایجاد دستی گره ها</p>		

مرحله بعد از تعیین گره ها، تعیین لبه ها می باشد. برای تعیین لبه ها می توان از منوی اصلی گزینه edge را انتخاب نمود.




شکل ۴۷: انتخاب منوی Edge از منوی اصلی

بعد از انتخاب edge از منوی اصلی پنجره زیر ظاهر می شود.



شکل ۴۸: پنجره ظاهر شده بعد از انتخاب edge


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

مشخصات فایل را در فایلی با پسوند edge.xml. (به عنوان مثال ex_EDGE_edg.xml) ذخیره کنید.

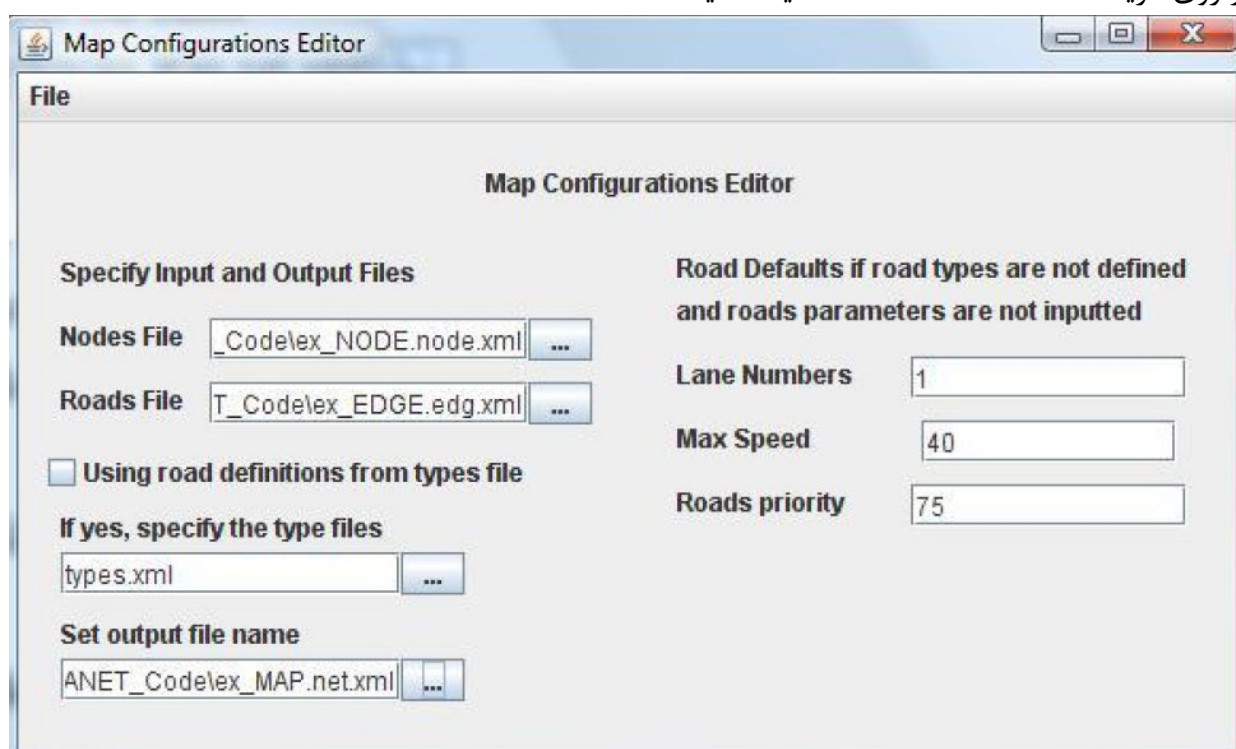
جزئیات مثال در زیر آمده است.

```
<edges>
<edge id="edgeR-0-0" fromnode="node1" tonode="node2" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeR-0-1" fromnode="node2" tonode="node3" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeL-0-0" fromnode="node2" tonode="node1" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeL-0-1" fromnode="node3" tonode="node2" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeU-0-0" fromnode="node1" tonode="node4" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeD-0-0" fromnode="node4" tonode="node1" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeU-0-1" fromnode="node2" tonode="node5" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeD-0-1" fromnode="node5" tonode="node2" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeU-0-2" fromnode="node3" tonode="node6" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeD-0-2" fromnode="node6" tonode="node3" priority="75"
nolanes="2" speed="40" length="200"/>
<edge id="edgeR-1-0" fromnode="node4" tonode="node5" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeR-1-1" fromnode="node5" tonode="node6" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeL-1-0" fromnode="node5" tonode="node4" priority="75"
nolanes="2" speed="40" length="300"/>
<edge id="edgeL-1-1" fromnode="node6" tonode="node5" priority="75"
nolanes="2" speed="40" length="300"/>
```

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre> <edge id="edgeU-1-0" fromnode="node4" tonode="node7" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeD-1-0" fromnode="node7" tonode="node4" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeU-1-1" fromnode="node5" tonode="node8" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeD-1-1" fromnode="node8" tonode="node5" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeU-1-2" fromnode="node6" tonode="node9" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeD-1-2" fromnode="node9" tonode="node6" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeR-2-0" fromnode="node7" tonode="node8" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeR-2-1" fromnode="node8" tonode="node9" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeL-2-0" fromnode="node8" tonode="node9" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeL-2-1" fromnode="node9" tonode="node8" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeU-2-0" fromnode="node7" tonode="node10" priority="75" nolanes="2" speed="40" length="200"/> <edge id="edgeD-2-0" fromnode="node10" tonode="node7" priority="75" nolanes="2" speed="40" length="200"/> <edge id="edgeU-2-1" fromnode="node8" tonode="node11" priority="75" nolanes="2" speed="40" length="200"/> <edge id="edgeD-2-1" fromnode="node11" tonode="node8" priority="75" nolanes="2" speed="40" length="200"/> <edge id="edgeU-2-2" fromnode="node9" tonode="node12" priority="75" nolanes="2" speed="40" length="200"/> <edge id="edgeD-2-2" fromnode="node12" tonode="node9" priority="75" </pre>		

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
<pre> nolanes="2" speed="40" length="200"/> <edge id="edgeR-3-0" fromnode="node10" tonode="node11" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeR-3-1" fromnode="node11" tonode="node10" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeL-3-0" fromnode="node11" tonode="node12" priority="75" nolanes="2" speed="40" length="300"/> <edge id="edgeL-3-1" fromnode="node12" tonode="node11" priority="75" nolanes="2" speed="40" length="300"/> </edges> </pre>		
<p>جزئیات لبه های تعیین شده</p>		

مرحله بعد، تعیین پیکربندی گره ها در نرم افزار SUMO است. مرحله بعد پیکربندی شبکه است باید بر روی گزینه CONFIGURATION کلیک کنید.



شکل(۱). پنجره باز شده بعد از کلیک کردن بر روی گزینه configuration

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

در این مرحله پیکربندی های لازم را برای تولید نقشه (MAP) انجام می دهیم. ادرس دو فایل حاوی گره ها و لبه ها و همچنین ادرس فایل نقشه (ex_MAP.net.xml) که باید در مرحله بعد تولید شود. سپس آن را با فایل netc.cfg ذخیره می کنیم. (در این مثال ex_MAP.netc.cfg) جزئیات فایل ex_MAP.netc.cfg در ادامه آمده است.

<pre> <configuration> <files> <xml-node-files>D:\Simulation\VANET_Code\ex_NODE.node.xml</xml-node-files> <xml-edge-files>D:\Simulation\VANET_Code\ex_EDGE.edg.xml</xml-edge-files> <xml-connection-files /> <type-file /> <output-file>D:\Simulation\VANET_Code\ex_MAP.net.xml</output-file> </files> <defaults> <type>Unknown</type> <lanenumber>1</lanenumber> <speed>40</speed> <priority>75</priority> <capacity-norm /> </defaults> <reports> <verbose /> <print-options /> </reports> <process> <remove-geometry>x</remove-geometry> </process> </configuration> </pre>	ex_MAP.netc.cfg خروجی فایل
---	----------------------------

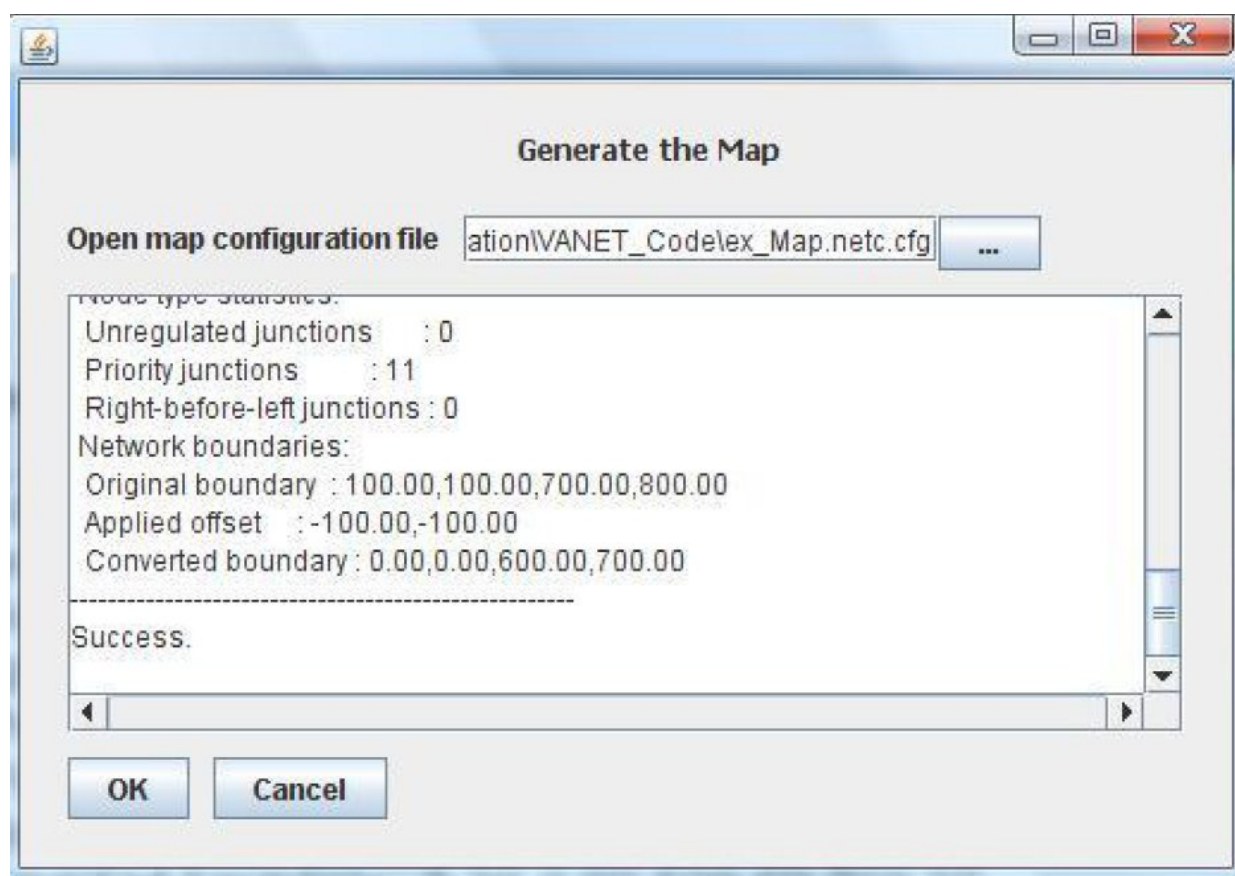
۵-۴-۲- تولید نقشه


در این مرحله نوبت به ایجاد نقشه ای که در مرحله قبل پیکربندی کردیم انجام می دهیم. بدین منظور بر روی create Map کلیک می کنیم.



شکل ۴۹: تولید نقشه

مسیر فایل ex_MAP.netc.cfg را که در مرحله قبل ایجاد شده است را وارد می کنیم و بر روی دگمه ok کلیک می شود. فایلی با پسوند net.xml (ex_MAP.net.xml) تولید خواهد شد که فایل نقشه مورد نظر است.

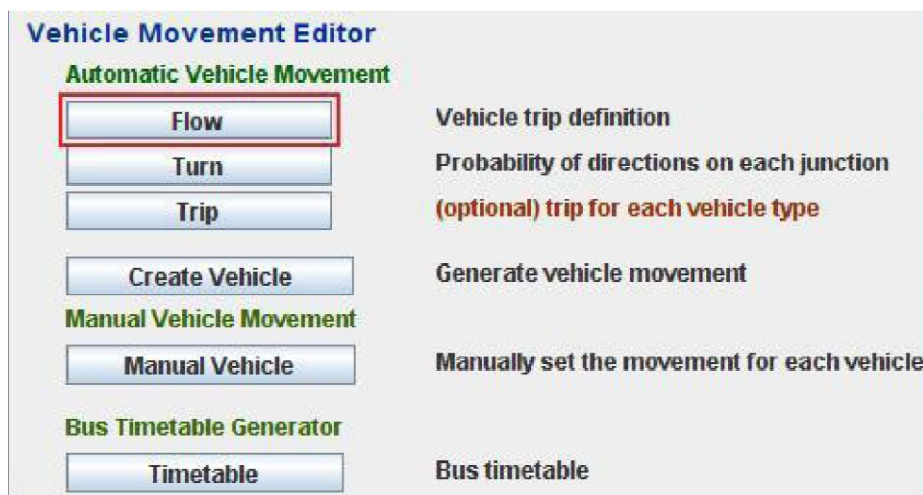


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

۵-۴-۵- تولید حرکت خودروها

۵-۴-۶- تعریف جریان

برای تعیین جریان حرکت خودروها بر روی Flow کلیک کنید.

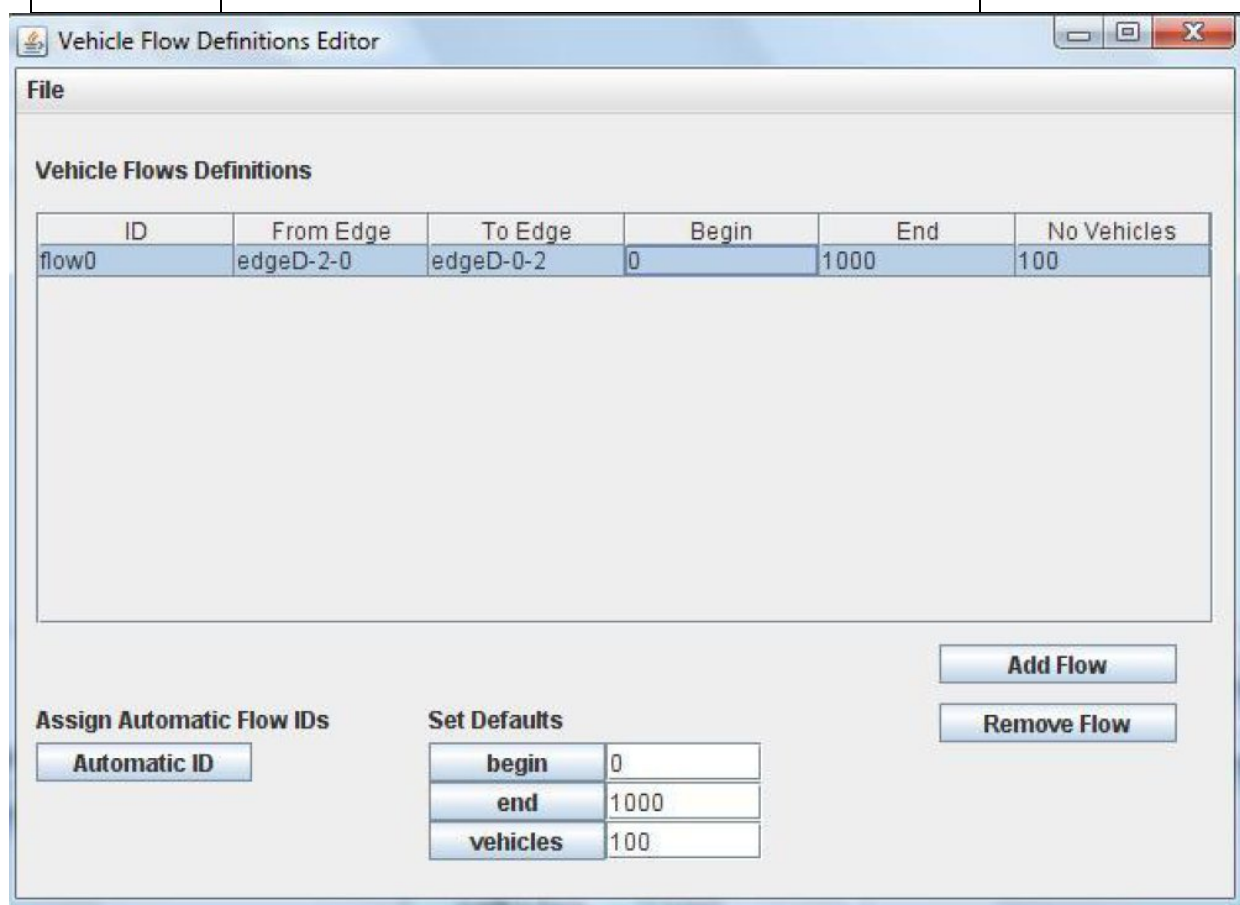


شکل ۵۰: پنجره ظاهر شده برای تعریف جریان

بعد از کلیک بر روی flow ، پنجره زیر ظاهر می شود.

دانشگاه علم و صنعت ایران
مرکز آموزش های الکترونیکی


عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2



شکل ۵۱: پنجره ظاهر شده پس از کلیک کردن بر روی flow

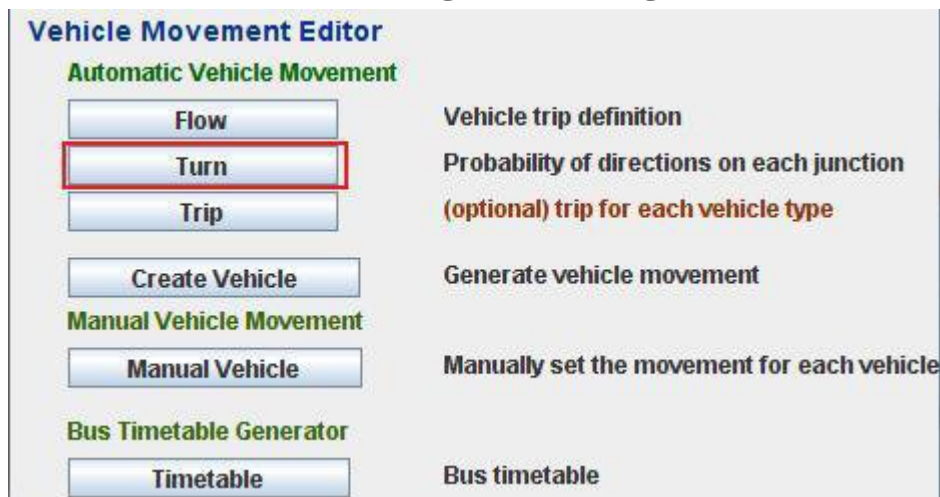
در این مثال حرکت گروهی خودروها از لبه D-2-0 به لبه D-0-2 است. در تعریف جهت حرکت گره ها باید توجه کرد که اگر گره می خواهد یک خودرو را از گره ۱ به گره ۲ و گره ۳ حرکت دهد. باید جریان را از لبه ۱ به ۳ تعریف کرد. اگر حرکت خودرو از لبه ۱ به ۲ تعریف شود. نصف راه را حرکت خواهد کرد. تنظیمات را در فایل ex_FLOW.flow.xml ذخیره می شود.

```
<flows>
<flow id="flow0" from="39dged-2-0" to="39dged-0-2" begin="0" end="1000" no="100"
/>
</flows>
```


	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---


۵-۴-۷- تعیین دور

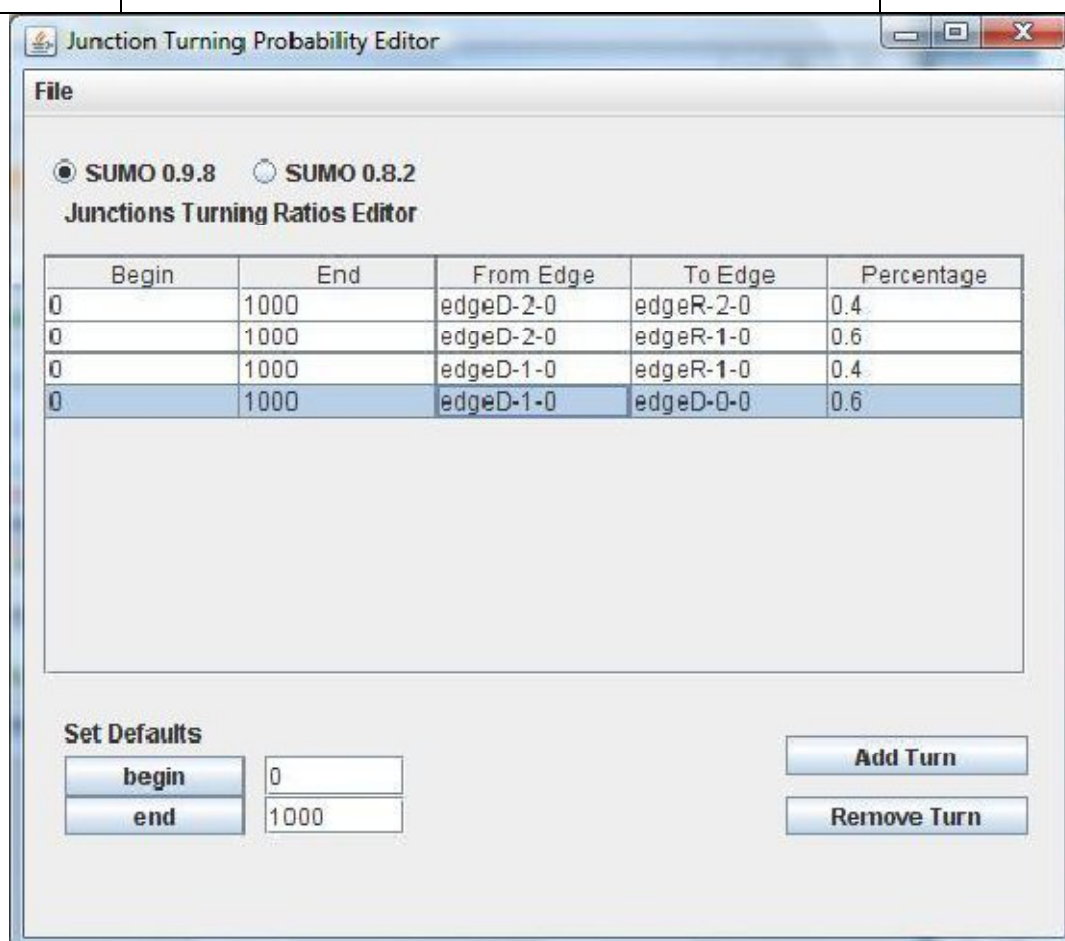
برای تعیین دور باید از منوی اصلی Turn انتخاب می شود.



شکل ۵۲: تعیین دور در منو اصلی

در این مرحله دور و سرعت آن ها را مشخص می کنیم.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---




شکل ۵۳: پنجره ظاهر شده پس از کلیک بر روی Turn

آنرا در فایل ec_TURN.turn.xml ذخیره می شود.

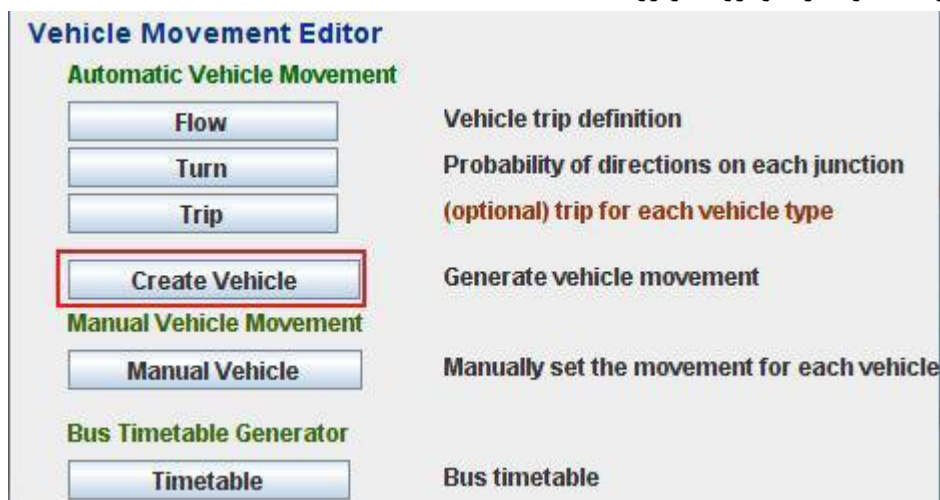
```
<turn-defs>
<interval begin="0" end="1000">
<fromedge id="40dged-2-0">
<toedge id="40dged-2-0" probability="0.4" />
<toedge id="40dged-1-0" probability="0.6" />
</fromedge>
<fromedge id="40dged-1-0">
<toedge id="40dged-1-0" probability="0.4" />
<toedge id="40dged-0-0" probability="0" />
</fromedge>
</interval>
</turn-defs>
```

فایل خروجی ec_TURN.turn.xml

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>

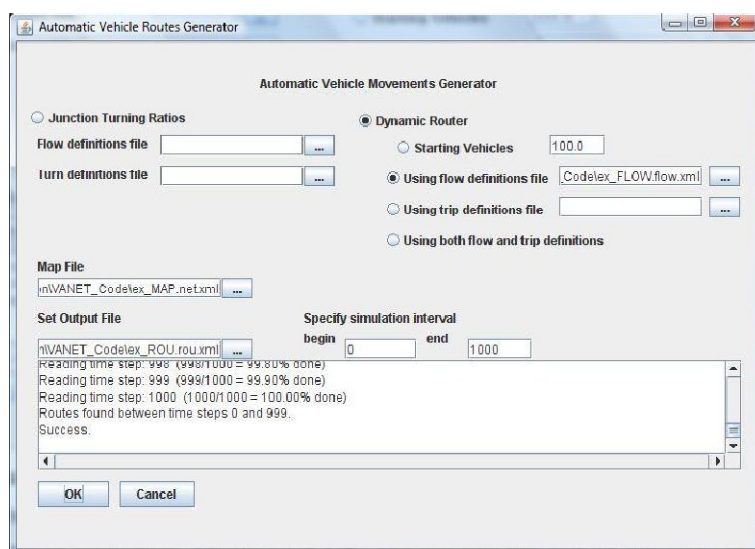
۵-۴-۸- حرکت خودکار خودروها

برای حرکت خودکار خودروها بر روی create vehicle کلیک کنید.



شکل ۵۴: حرکت خودکار خودروها

این مرحله گروهی از خودروها را در آغاز شبیه سازی ایجاد می کند.



شکل ۵۵: پنجره ظاهر شده پس از کلیک بر روی create vehicle

مسیر فایل های جریان ex.FLOW.flow.xml و نقشه ex_MAP.net.xml و همچنین مسیر فایل خروجی مسیرها ex_ROU.rou.xml را مشخص می کند.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

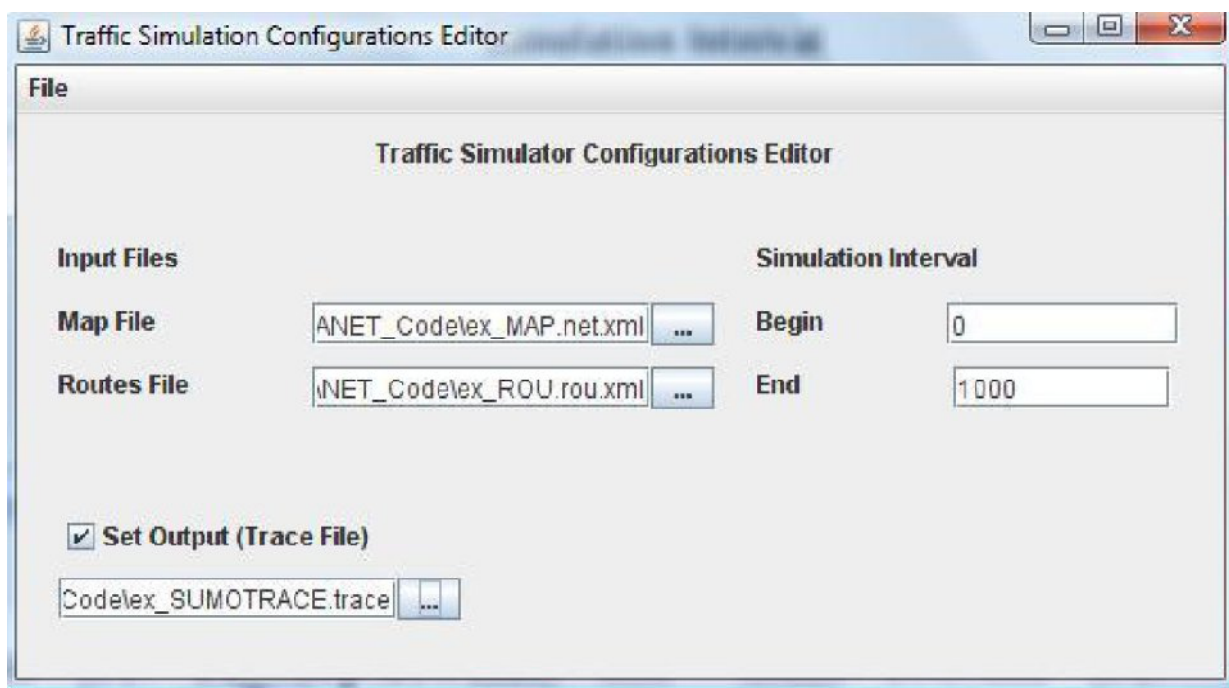
۵-۴-۹- تنظیمات شبیه سازی

در مرحله بعد به تنظیمات شبیه سازی اشاره می شود. بر روی گزینه configuration کلیک کنید.




شکل ۵۶: کلیک بر روی گزینه configuration

در این مرحله فایل مورد نظر sumo را تولید می کند. فایل های ex.MAP.net.xml و ex_ROU.rou.xml را وارد کنید. اگر به فایل trace در SUMO نیاز دارید. گزینه ان را انتخاب کنید و مسیر خروجی را به آن بدهید. (ex_SUMOTRACE.tr). سپس تنظیمات را در فایل ex_SUMO.sumo.cfg ذخیره کنید. پس از کلیک بر روی configuration پنجره زیر را می بینید.



شکل ۵۷: پنجره ظاهر شده پس از کلیک بر روی configuration

پس از این مرحله Visualization می شویم بر روی گزینه Visualization کلیک کنید.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

Simulation

Visualization

Configuration

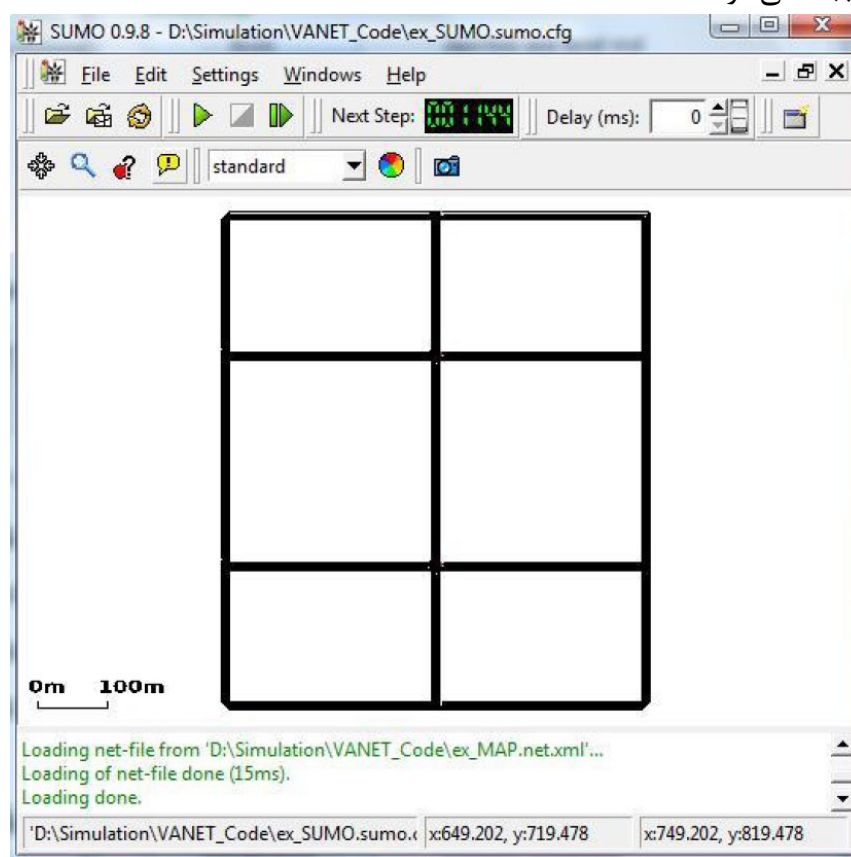
Run Simulation

Simulation configuration

Visualize simulation

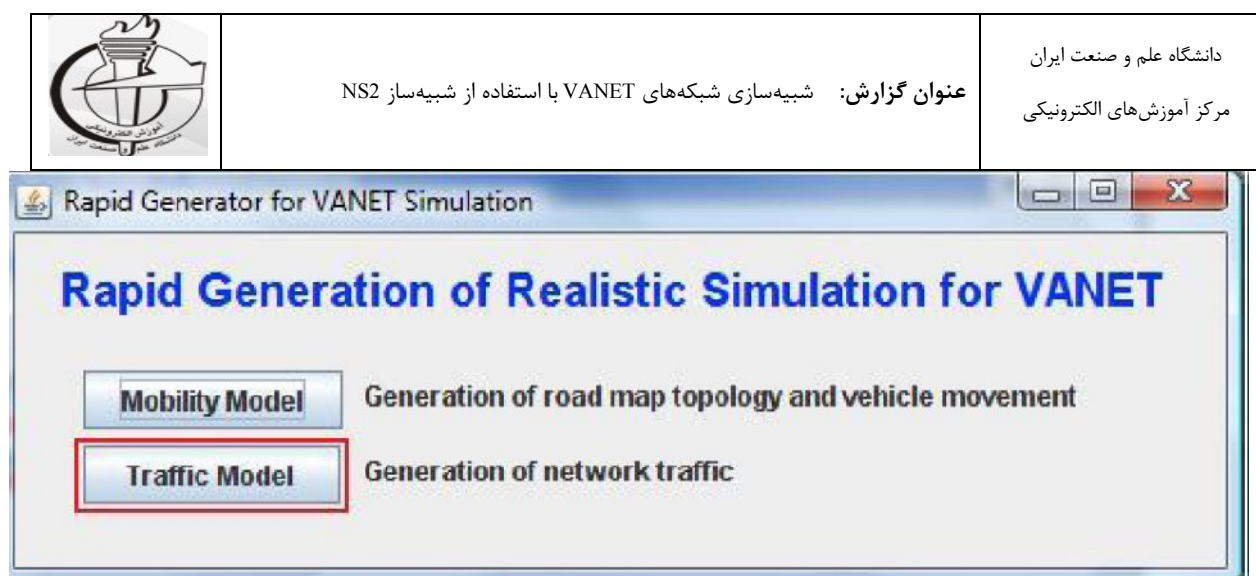
Run simulation on background

بعد از کلیک بر روی Visualization ، برنامه SUMO اجرا می شود و فایل ex.SUMO.sumo.cfg در محیط گرافیکی ایجاد می شود.



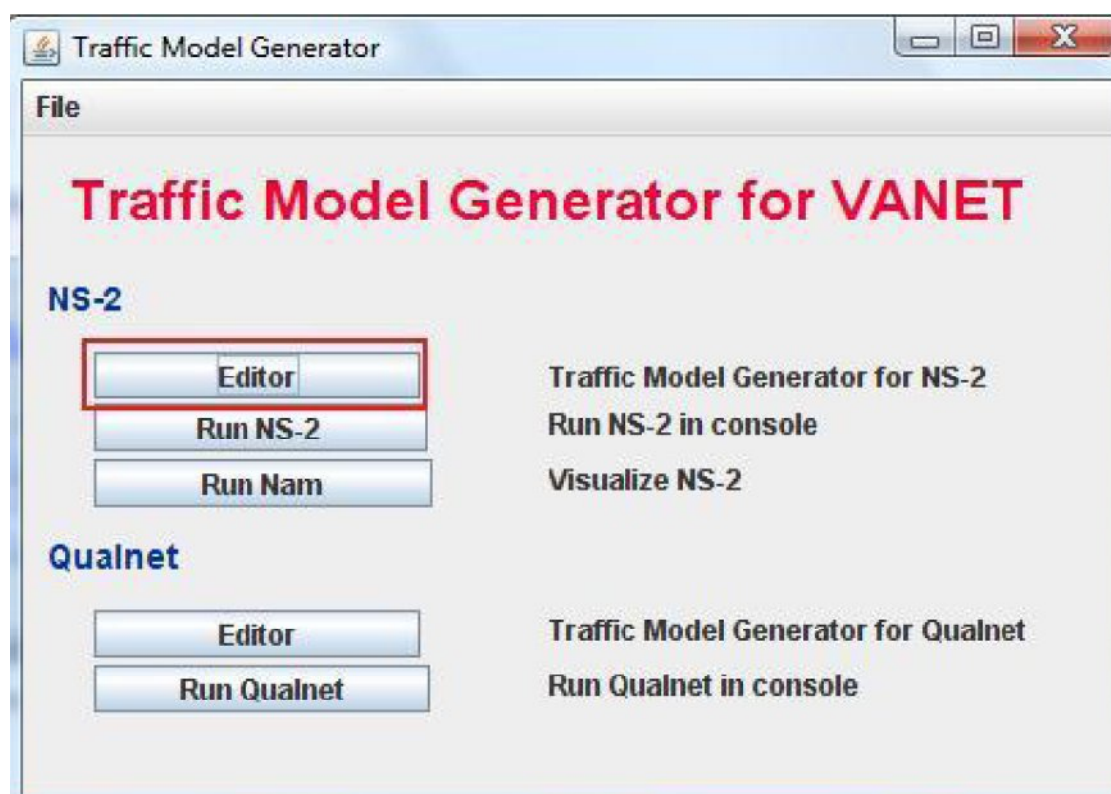
شکل ۵۸: پنجره ظاهر شده پس از کلیک بر روی Visualization

در مرحله آخر مدل ترافیکی تولید می شود و گزینه Traffic Model انتخاب می شود.




شکل ۵۹: انتخاب برای مدل ترافیک

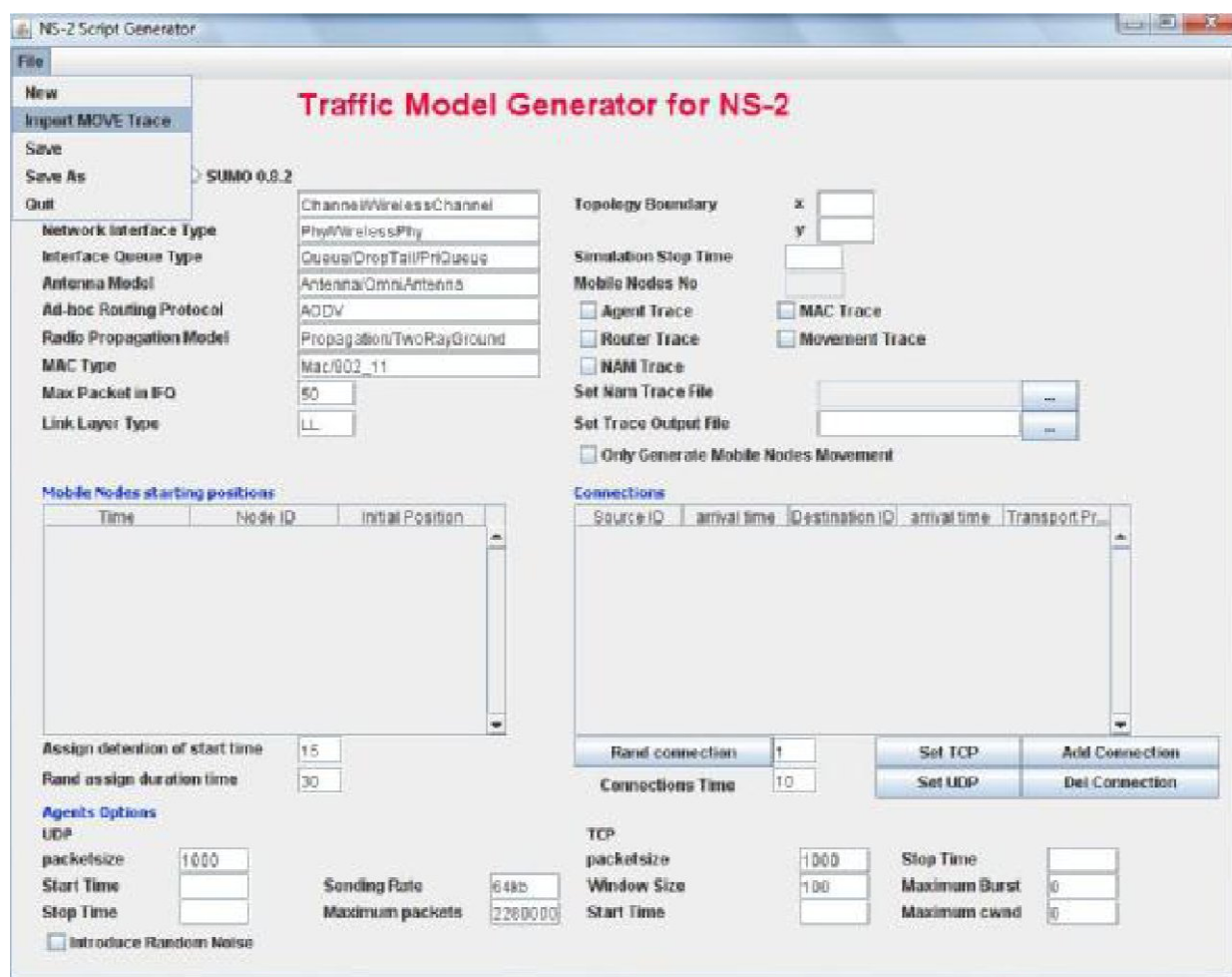
گزینه traffic model دارای دو بخش اصلی برای نرم افزار NS2 و Qualnet است که فقط به شبیه ساز ns2 اشاره می شود.



شکل ۶۰: مدل ترافیک نرم افزار ns2


این بخش فایل شبیه سازی ترافیکی یک فایل Tcl را برای ns2 ایجاد می شود. ابتدا فایل trace (ex_SUMOTRACE.trace) را برای SUMO که در مرحله قبل تولید شده است تولید می کنیم و پس از آن فایل نقشه (ex_MAP.net.xml) را وارد می کنیم. بسته به حجم فایل و سرعت کامپیوتر باید مدتی صبر کنید تا اطلاعات بار شود.

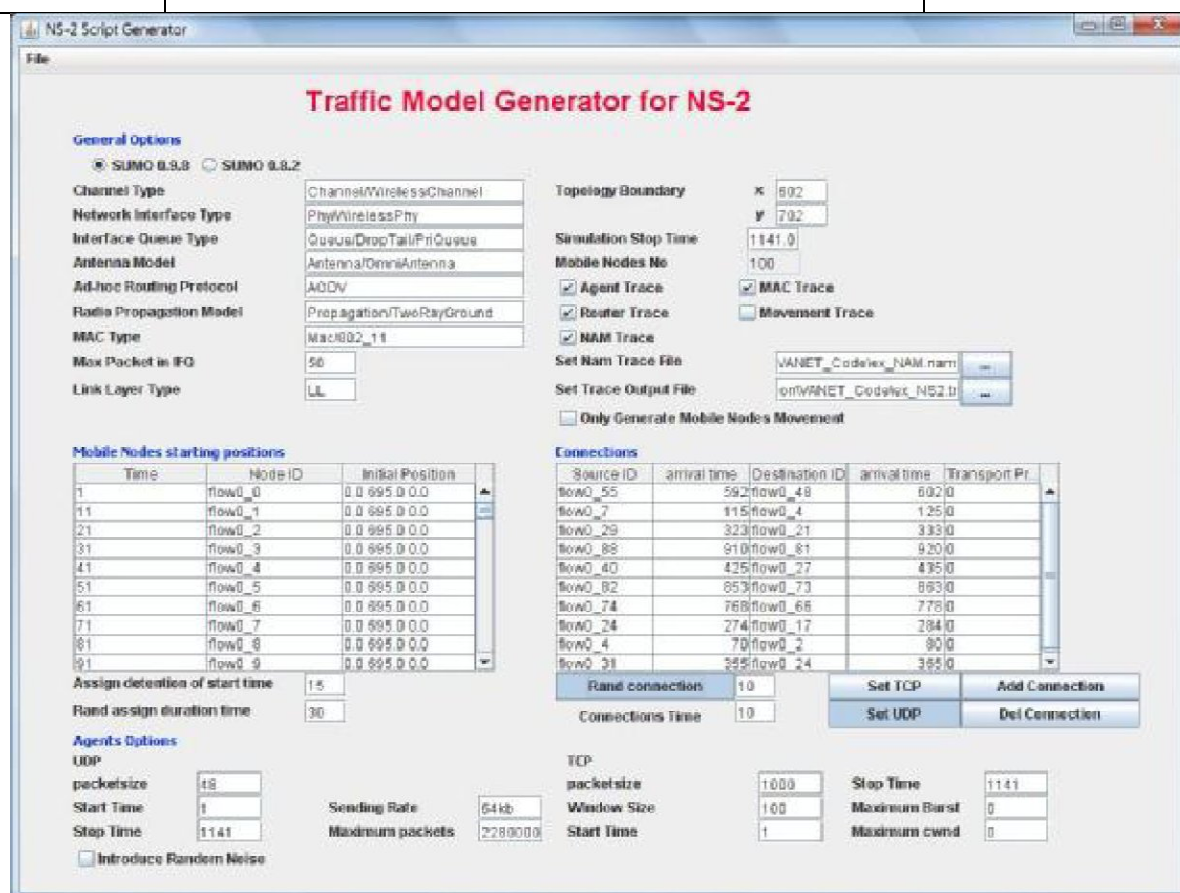
	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---



شکل ۶۱: خروجی برای ns2 تولید کردن

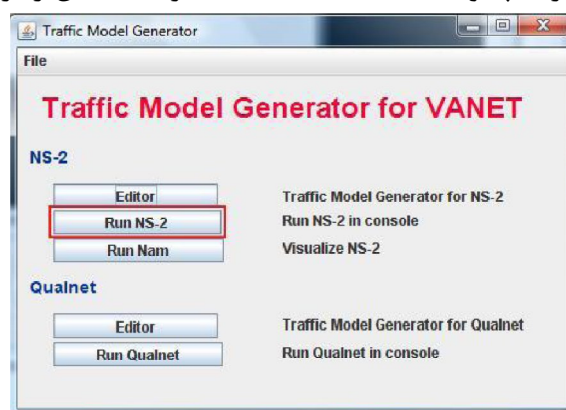
بعد از بارگذاری، گزینه مربوط به شبیه سازی ns2 را مشخص می کنیم. به عنوان مثال فایل trace را برای nam مشخص می کنیم و ارتباط بین گره ها و نوع آن tcp و udp را مشخص می کنیم.

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---



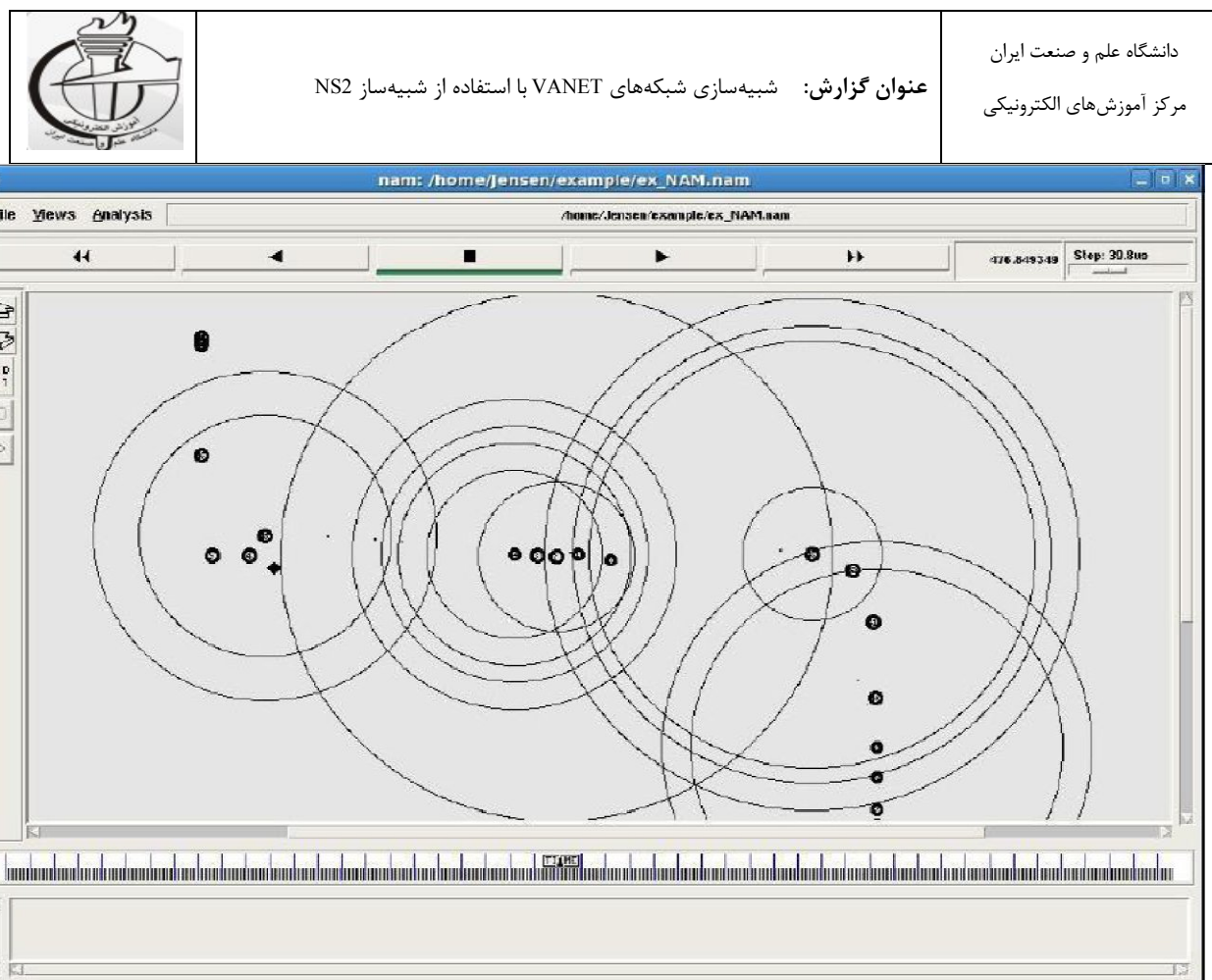
شکل ۶۲: وارد کردن اطلاعات مربوط به شبیه ساز ns2

بعد از انجام تنظیمات ان را با پسوند tcl. (ex_NS.tcl) ذخیره کنید. ان را از طریق NS2 اجرا کنید.




شکل ۶۳: وارد کردن اطلاعات در NS2

می توانید خروجی ان را در NAM ببینید.



شکل ۶۴: خروجی در شبیه ساز NAM

	<p>عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2</p>	<p>دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی</p>
--	--	---

منابع و مراجع

- <http://www.isi.edu/nsnam/ns/>
- <http://nile.wpi.edu/NS/>
- <http://www.isi.edu/nsnam/nam/>
- <http://vanet.info/node/13>
- Marco Fiore, "Mobility Models in Inter-Vehicle Communications Literature", Technical Report, November 2006.
- Vehicular Ad hoc Networks, Master's Thesis in Computer Science Rainer Baumann, ETH Zurich 2004
- Christoph Sommer and Falko Dressler, The DYMO Routing Protocol in VANET Scenarios, IEEE 2007

	عنوان گزارش: شبیه سازی شبکه های VANET با استفاده از شبیه ساز NS2	دانشگاه علم و صنعت ایران مرکز آموزش های الکترونیکی
--	---	---

