doi:10.3772/j.issn.1002-0470.2013.08.005

基于自适应 time step 算法的网络模拟流量模型研究①

孙 越②*,** 郝志宇③* 云晓春* 费海强*

(*中国科学院信息工程研究所 北京 100093)

(**北京邮电大学电子工程学院 北京 100876)

摘 要 通过对网络模拟流量模型的分析,深入研究了基于时间驱动流量模型的流模拟方式,并针对目前基于时间驱动的流量模型 time step 选取方法存在的缺陷,提出一种根据网络行为的复杂程度不断修改 time step 的自适应 time step 选取策略。实验数据表明,相比于目前流量模型 time step 选取策略,这种新的选取策略大幅度降低了网络模拟的执行时间,同时提高了模拟的精确度,对大规模的网络模拟也具有很好的效果,为建设大规模高真实性的实时网络模拟平台奠定了基础。

关键词 网络模拟,流量模型,自适应龙格库塔算法,PRIME 平台,性能优化

0 引 言

随着因特网规模的不断扩大,如何更好地设计网络结构和协议,已成为人们研究的重点。目前的研究方法主要有网络模拟、数学建模等[1,2]。网络模拟是研究网络行为与网络协议的重要方法。网络模拟因其真实性较高、模拟规模较大、开发和运行成本较低等优点而备受青睐。目前很多网络模拟器都采用包级别的模拟。虽然包级别的模拟能保证较高的真实性,但对模拟器的计算开销来说是极大的挑战,由此引入了流量模型。在网络模拟研究中,流量模型是提高网络模拟性能的工具,特别是基于时间驱动的流量模型在模拟效率上有很大优势。目前的流量模型大都采用流模拟方式,即将具有相同属性的数据包抽象为一个流,在模拟过程中更新某些属性的值,以此来描述网络特征,该方式可降低单位时间内模拟器处理事件的数量。流模拟方式主要分为基于流的事件驱动模拟和基于流的时间驱动模拟[3]。基于流的事件驱动模拟[4,5]是当数据流大小变化时,即产生一个离散事件,用以触发此数据流相关属性的变化[4,5]。这种模拟适用于数据流变化不大的情况,如果数据流变化剧烈,此方法性能受损严重,且易引发连锁反应;基于流的时间驱动模拟是将模拟时间周期性地向前推进一定的时间间隔(time step)[6-9]。每次推进都计算一次网络状态对流量模型的反馈并修改流量模型的参数[5-8]。由于基于流的时间驱动模拟在效率上有较大优势,因而它将成为流量模型的发展趋势[3]。

对于基于流的时间驱动模拟,有效地选取 time step 是保证流量模型高效和高真实性的关键。目前 time step 的选取缺乏有效的解决方案,采取固定值或大小两个值来设置 time step[8],在精确度和效率上都有一定缺陷。Li 等提出了一种动态选取 time step 的算法[9],在效率上有了改善,但其算法在精确度上仍需改进。针对以往 time step 选取策略存在的问题,本文提出了一种自适应 time step 算法,该算法根据网络行为的变化实时更新 time step,使变化保持在允许的精度范围内,从而有效地提升流量模拟的效率和真实性。

1 流量模型

Misra 等提出的流量模型(也称 MGT 模型)是利用一组常微分方程来描述 TCP 窗口大小、队列长度和丢包率等网络属性[7-10]。在流量模型中,表示

① 国家自然科学基金(61003261)资助项目。

② 女,1985 年生,博士生;研究方向:保密通信;联系人,E-mail:sunyue1101@bupt.edu.cn

③ 通讯作者,E-mail:haozhiyu@iie.ac.cn

(收稿日期:2012-09-10)

同源同目的的一组流量,通常叫做一个类(Class)。在 t 时刻,流量模型 TCP 拥塞窗口 $W_i(t)$ 的加性增长和乘性减小特性满足方程

$$\frac{\mathrm{d}W_i(t)}{\mathrm{d}t} = \frac{1(W_i(t) < M_i)}{R_i(t)} - \frac{W_i(t)}{2}\lambda_i(t) \quad (1)$$

其中,M_i 为最大窗口值,并且 $0 \leqslant W_i(t) < M_i$,$R_i(t)$ 为往返延时,$\lambda_i(t)$ 为同源同目的流的丢包速率。

瞬时队列长度 $q_l(t)$ 可以表示为

$$\frac{\mathrm{d}q_l(t)}{\mathrm{d}t} = \xi_l(t)(1 - p_l(t)) - C_l \quad (2)$$

其中,$0 \leqslant q_l(t) < Q_{max}$,$\xi_l(t)$ 为链路 l 的聚合到达速率并且 $\xi_l(t) = \sum_{i \in \mathbf{N}_l} A_i^l(t)$,$p_l(t)$ 为 t 时刻链路 l 上的丢包率,C_l 为链路上带宽。

平均队列长度 $x_l(t)$ 与瞬时队列长度 $q_l(t)$ 满足关系式

$$\frac{\mathrm{d}x_l(t)}{\mathrm{d}t} = \frac{\ln(1-\alpha)}{h}[x_l(t) - q_l(t)] \quad (3)$$

其中,h 为龙格库塔算法步长,通常为一个定值。α 是计算 RED 队列中 EWMA 的权重。

丢包率 $p(x)$ 可以根据平均队列长度 $x_l(t)$ 求出,通常情况下满足以下线性分段函数:

$$p(x) = \begin{cases} 0 & (0 \leqslant x < q^{\min}) \\ \dfrac{x - q^{\min}}{q^{\max} - q^{\min}} p^{\max} & (q^{\min} \leqslant x < q^{\max}) \\ \dfrac{x - q^{\max}}{q^{\max}}(1 - p^{\max}) + p^{\max} & (q^{\max} \leqslant x < 2q^{\max}) \end{cases} \quad (4)$$

其中 $q^{\min}$,$q^{\max}$ 和 $p^{\max}$ 是 RED 策略的参数,若 $q^{\min} = q^{\max} = Q_l$,则 RED 策略变为 drop-tail 策略。

2　现有算法回顾及存在的问题

针对描述流量模型的常微分方程,需要确定每个时刻拥塞窗口、队列长度、丢包率等网络特性的精确解。以拥塞窗口方程为例,TCP 支持慢启动的算法,初始拥塞窗口为 1 个报文段,由常微分方程(1)及初始条件:

$$\begin{cases} \dfrac{\mathrm{d}W_i(t)}{\mathrm{d}t} = \dfrac{1}{R_i(t)} - \dfrac{W_i(t)}{2}\lambda_i(t) \\ W_i(0) = 1 \end{cases} \quad (5)$$

拥塞窗口精确解 $W_i(t)^*$ 如下:

$$W_i(t)^* = \frac{2 + (\lambda_i(t)R_i(t)e^{-\frac{\lambda_i(t)}{2}t} + 2)}{R_i(t)\lambda_i(t)} \quad (6)$$

但并非每个常微分方程都能顺利求解,而且在网络模拟平台上嵌入求解微分方程的模块,会加大模拟平台的计算开销。因此需要引入数值方法近似地求解,目前最常用的是龙格库塔算法。

2.1　龙格库塔算法

利用龙格库塔算法解决流量模型通常选取固定 time step 值,即将模拟过程按照时间分成很小且等距的 time step,在每一个 time step 中求解流量模型的常微分方程,并不断地更新网络参数。具体过程是将常微分方程(1)—(3)化为一阶微分的一般形式[11-14]:

$$\frac{\mathrm{d}y_i(x)}{\mathrm{d}x} = f_i(x, y_1, y_2, \cdots, y_N), \quad i = 1, 2, \cdots, N \quad (7)$$

其中 f_i 是已知函数,利用经典的四阶龙格库塔公式(式中 h 为本文中的 time step):

$$\begin{cases} y_{n+1} = y_n + \dfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 = f(x_n, y_n) \\ K_2 = f(x_n + \dfrac{h}{2}, y_n + \dfrac{h}{2}K_1) \\ K_3 = f(x_n + \dfrac{h}{2}, y_n + \dfrac{h}{2}K_2) \\ K_4 = f(x_n + h, y_n + hK_3) \end{cases} \quad (8)$$

在每个 time step 中对右端进行 4 次计算,可以逐阶地消除误差项。

但是随着网络行为复杂性的增加,如果仅依靠固定的 time step 值来计算,就会导致计算误差的增大。当 time step 选取过短,而网络变化并不大时,会导致流量模型的更新过于频繁,降低模拟性能;而当 time step 过长,模拟流量变化剧烈时,则会影响模拟的真实性。

2.2　LT time step 算法

Li Ting 等针对固定 time step 方法存在的问题,提出了动态选取 time step 的算法,即在模拟过程中网络变化较陡时减小 time step,反之加大 time step 以加快计算过程[9]。该算法(称 LT time step 算法)以队列长度的变化量 Δq 作为观测对象,引入参数 $\tau = hC_l$,其中 h 为当前时刻的 time step,C_l 为包平均大小/带宽。具体算法如下:如果队列长度的变化量 Δq 变化大于 2τ,则下一时刻 time step 值变为原来的 $\tau/\Delta q$ 倍;如果 Δq 变化小于 0.5τ,则 time step 扩大 1 倍;否则 time step 保持不变。

该算法实现了一定的自适应性,但也存在一些问题:首先,算法中 τ 是个小量,Δq 的变化相对于 τ

较大,将 Δq 与 τ 比较的结果来作为判断网络行为的变化情况有些不合适;其次,通过分析实验数据,在 Δq 为 $10^3 \sim 10^4$ 时,time step 值始终为原始 time step,并非每步都更新;再次,该算法仍然是基于四阶龙格库塔算法,所以在精确度上还有待提升。

3 自适应龙格库塔算法的引入与实现

3.1 自适应龙格库塔算法

针对固定 time step 龙格库塔算法在计算上的局限性,自适应龙格库塔算法根据被积函数在积分区间上的变化陡缓来自动确定每个子区间的步长。Felhberg 提出了一种 4 阶龙格库塔算法和 5 阶龙格库塔算法组合的格式[11]:

$$\begin{cases} y_{n+1} = y_n + h(\frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4101}K_4 - \frac{1}{5}K_5) + O(h^6) \\ \hat{y}_{n+1} = y_n + h(\frac{16}{135}K_1 + \frac{6656}{12825}K_3 + \frac{28561}{56430}K_4 - \frac{9}{50}K_5 + \frac{2}{55}K_6) + O(h^5) \\ K_1 = f(x_n, y_n) \\ K_2 = f(x_n + \frac{h}{4}, y_n + \frac{h}{4}K_1) \\ K_3 = f(x_n + \frac{3h}{8}, y_n + \frac{3}{32}K_1 + \frac{9}{32}K_2) \\ K_4 = f(x_n + \frac{12h}{13}, y_n + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3) \\ K_5 = f(x_n + h, y_n + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4) \\ K_6 = f(x_n + \frac{h}{2}, y_n - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5) \end{cases} \tag{9}$$

此格式的截断误差为

$$T_{n+1} = y_{n+1} - \hat{y}_{n+1} \tag{10}$$

由式(9),y 与 h^5 成正比,令前一时刻 time step 为 h_1,下一步最佳 time step 为 h_0,Δ_0 为所需精度的一个向量($\Delta_0 = \varepsilon h$),ε 称为相对精度,由实际需要来设置。最佳步长 h_0 与 h_1 满足关系式[14]

$$h_0 = s \cdot h_1 \tag{11}$$

其中,

$$s = \begin{cases} \left|\frac{\Delta_0}{T_1}\right|^{0.20}, & \Delta_0 \geqslant T_1 \\ \left|\frac{\Delta_0}{T_1}\right|^{0.25}, & \Delta_0 < T_1 \end{cases} \tag{12}$$

这种方法是根据所设精度和观测对象变化大小来判断下一步 time step 增加还是减小。举例来说,以队列长度的变化 Δq 为参考,初始 time step 为 0.01s,令前一时刻 $\Delta q = q_l(t) - q_l(t-h) = 1000$,$\Delta q$ 的允许误差 $\varepsilon = 10^4$,则精度向量 $\Delta_0 = \varepsilon h = 100$,即 Δq 大于所设精度的向量,则 $s = |100/1000|^{0.25} = 0.56$,即减小 time step 为 0.0056s。对比前面介绍的几种 time step 选取策略,固定 time step 值始终为 0.01s;LT time step 算法的实验数据显示当 Δq 为 400 到 2000 之间时,time step 也始终保持 0.01s。由此可知,在网络流量变化比较大时,固定 time step 算法和 LT time step 算法都不能保持每步 time step 更新,而自适应 time step 算法则会实时地修改 time step,以达到网络模拟的高精确性和高效性。

3.2 自适应 time step 流量模型实现方案

在实际网络模拟平台的实现上,出于经验性的考虑,我们取式(12)中 $s < 0.75$ 时,折半步长,$s > 1.5$时,加倍步长,原因有以下 4 方面:

(1)可以防止积分步长过小或者过大,例如 $s = 1^{-10}$或者 1^{10}都会导致积分病态;

(2)从计算效率、收敛性、稳定性及合理性考虑,在 s 为 0.5 ~2 时比较容易确定出收敛速度;

(3)$s < 0.75$ 与 $s > 1.5$ 的选择和采用黄金分割步长($s = 0.618$;$s = 1/(1-0.618)$)具有相似的收敛速度,并且这样的选择也不会对积分精度和效率有多大损失;

(4)有数学上自适应龙格库塔算法的理论支撑,$s < 0.75$ 与 $s > 1.5$ 的选择不会增大计算量,即占用内存资源与其他 time step 算法没有大的差异。

算法的流程图如图 1 所示。

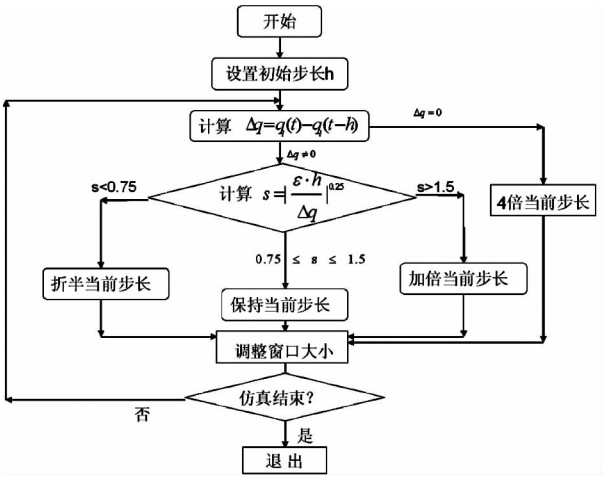


图 1 自适应 time step 算法流程图

4 实验测试及结果分析

4.1 实验平台介绍

实验利用的网络平台 PRIME 是基于 SSF(Scalable Simulation Framework)框架的网络模拟平台,由

C/C++编程实现[15]。PRIME 由 PRIME SSF 和 PRIME SSFNET 两部分组成,PRIME SSFNET 是建立在 PRIME SSF 上层的网络模拟器。本文对流量模型的研究就是基于 SSFNET 环境,使用 DML 语言来描述网络行为。

4.2　实验模型

实验仪器配置情况:酷睿双核 CPU E7300-2.66 GHz 处理器,1GBRAM。由于哑铃型网络结构具有一定的通用性,不妨以哑铃型网络结构为例(如图 2),两个网络 Net0 和 Net1 分别由 Client(S1,S2,S3)和 Server(D1,D2,D3)组成,三组 Class 分别为 S1→D1,S2→D2,S3→D3,每个 Class 含有 20 个同源同目的 TCP 流,中间由两个路由器 R 链接且 R-R 带宽为 10Mbps,S(D)-R 带宽为 100Mbps,传输延时为 0.02s,链路延时为 0.025s,路由采取 RED 策略,参数配置见表 1。

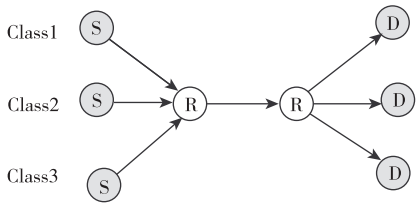


图 2　网络拓扑图

表 1　RED 网络配置参数

weight	q^{min}	q^{max}	p^{max}	buf	pktsize
0.01	8e5	3.6e7	0.2	5MB	4000

设置初始 time step 为 0.001s,三类 TCP 发送数据流时间如下:

Class 1:0s-100s;

Class 2:0s-40s,70s-100s;

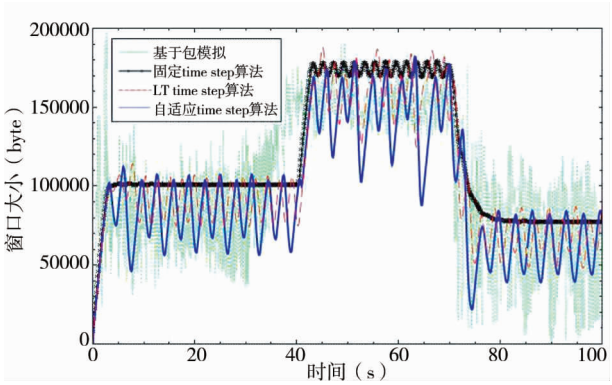
Class 3:70s-100s。

我们进行 4 组实验:基于包级别的模拟、固定 time step 算法、LT time step 算法和本文的自适应 time step 算法,分别对 Class 1,Class 2 和 Class 3 的 TCP 拥塞窗口大小进行对比分析。

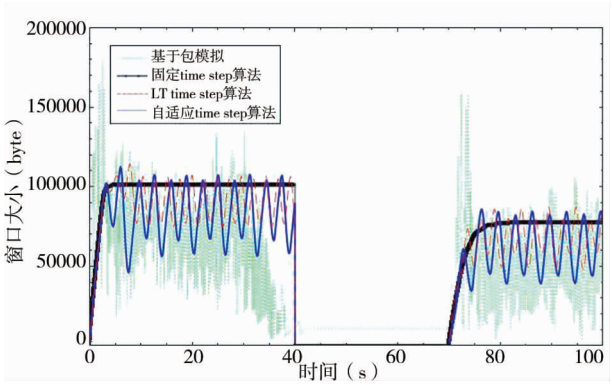
4.3　实验结果分析

4.3.1　精确度分析

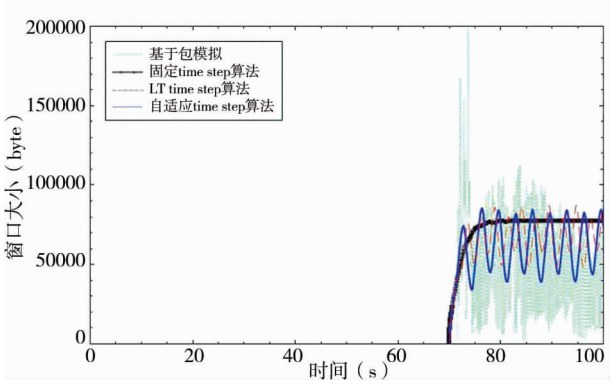
通过第 2、3 节对三种 time step 选取算法的描述,固定 time step 算法在精确度上是较差的,自适应 time step 算法是三种算法中效果最好的。图 3 验证了这一事实(暂不考虑基于包模拟的结果在初始 5s 有一个不稳定的冲击)。固定 time step 算法相比于基于包模拟的模拟结果偏高,并且在图 3(a)中 40s~70s 流量较复杂时,不能很好地反映流量复杂程度。LT time step 算法在一定程度上反映了流量的复杂程度,但与自适应 time step 算法的模拟结果相比,自适应 time step 算法的结果更加逼近于基于包模拟的结果而且能够较好地反映出复杂的流量变化。所以,自适应 time step 的流量模型在精确度方面比固定 time step 算法和 LT time step 算法效果更好。



(a)Class 1



(b)Class 2

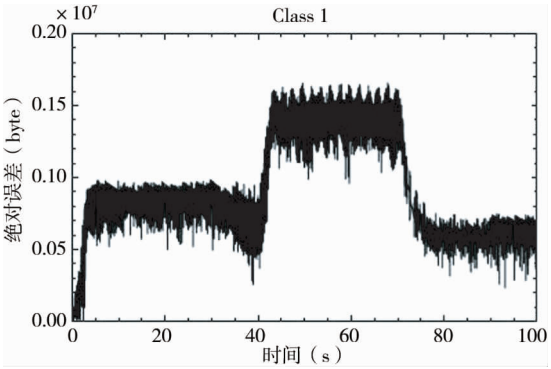


(c)Class 3

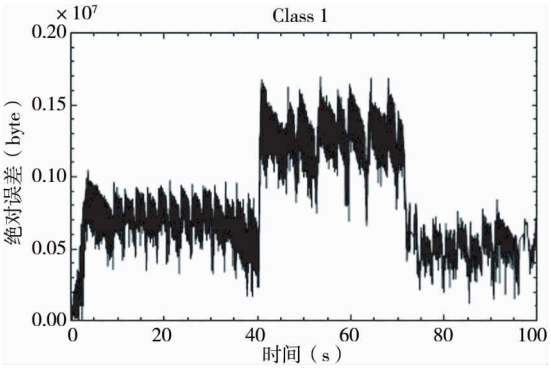
图 3　四种算法下 Class1-3 的 TCP 拥塞窗口大小的对比

4.3.2 误差分析

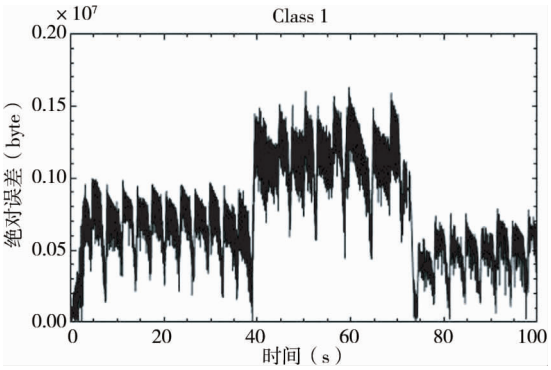
将实验中 Class 1 在固定 time step 算法、LT time step 算法和自适应 time step 算法下 TCP 窗口大小的结果，分别与基于包模拟算法的 TCP 窗口大小结果求绝对误差的分析图，如图 4 所示。



（a）固定 time step 算法



（b）LT time step 算法



（c）自适应 time step 算法

图 4 Class 1 在三种 time step 算法下的模拟结果与基于包模拟的结果的绝对误差的对比

由于自适应 time step 算法是基于 5 阶的龙格库塔算法，在精度上要高于其他的算法，图 4 表明自适应 time step 算法的模拟结果与基于包模拟的结果绝对误差比前两种算法更低。并进一步对图 4 中（a）、（b）、（c）三组绝对误差进行平均得到图 5，可知基于自适应 time step 算法的绝对误差比固定 time step 算法大约小 20%，比 LT time step 算法小 10% 左右。

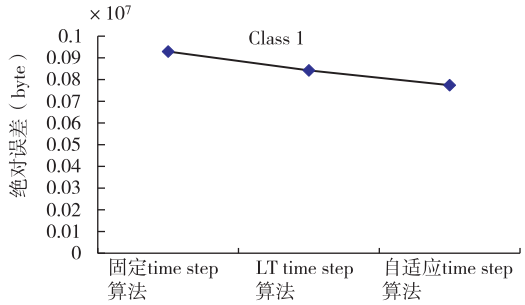


图 5 Class 1 在三种 time step 算法下的模拟结果与基于包模拟的结果的绝对误差值

4.3.3 效率分析

将图 2 网络模型中 Class 的数目分别调整为 3、5 和 10 组，每一组 Class 从 0s 到 100s 持续均匀地发送 20 个同源同目的 TCP 流量，分别比较在不同 TCP 流量负载的情况下，三种 time step 算法的执行时间，结果如图 6 所示。

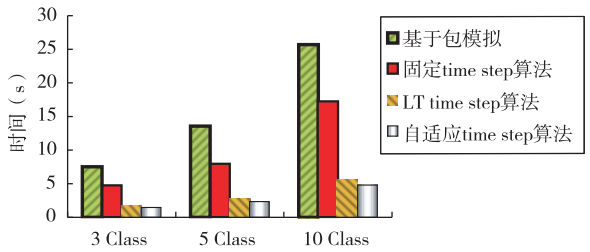


图 6 不同 TCP 流量负载情况下四种模拟算法的执行时间

由图 6 可知，在相同流量负载的情况下，基于自适应 time step 算法的执行时间比基于包模拟省时约 80%，比固定 time step 算法节省 70%，并且比 LT time step 算法省时 10% 左右；随着流量负载增大，这三种算法的执行时间都相应延长，但自适应 time step 算法仍然是执行时间最短且效率最高的。

4.4 大规模网络模拟实验

鉴于自适应 time step 算法对简单网络拓扑实验的可行性与有效性，下面将该算法应用于大规模网络进行模拟实验。

4.4.1 大规模网络模拟拓扑图

以 campus 网络模型为例（如图 7），每一个 campus 网络包含 4 个子网：Net 0，1，2，3，其中 Net 1 中

1:2—1:5 为 4 台 Server,其余子网中共有 12 个 Lan(椭圆表示,每个 Lan 含 42 个 client 和 1 个 router),包含 504 个 client 和 30 个 router,也就是说,一个 campus 含有 538 个网络节点。

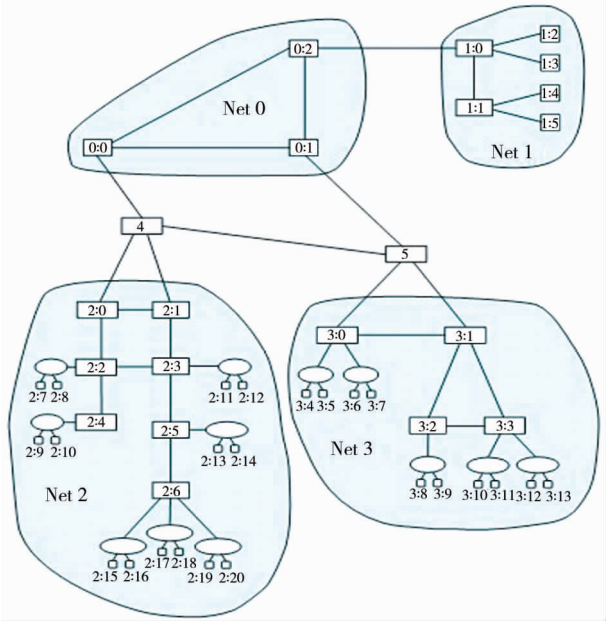


图 7　campus 网络拓扑图

将 1 个 campus 网络扩大规模至 2 个和 4 个 campus 组成的网络模型,即分别有 1076 和 2152 个网络节点。每个 campus 参数配置如表 2 所示。

表 2　campus 网络参数

	R2R	Lan	As2As
带宽	1Gbs	100Mbs	2Gbs
延时	0.005s	0.01s	0.2s
filesize		3×10^7	

4.4.2　大规模网络模拟结果分析

图 8 表明自适应 time step 算法在大规模网络模拟中同样取得了很好的效果,与基于包模拟和另外两种 time step 算法相比,执行效率最高。并且自适应 time step 算法在 1 个、2 个和 4 个 campus 组建的网络中比基于包模拟的结果分别省时 84.9%、85.8% 和 86.1%。因此,随着网络规模的不断扩大,这种效率上的优势更为明显。

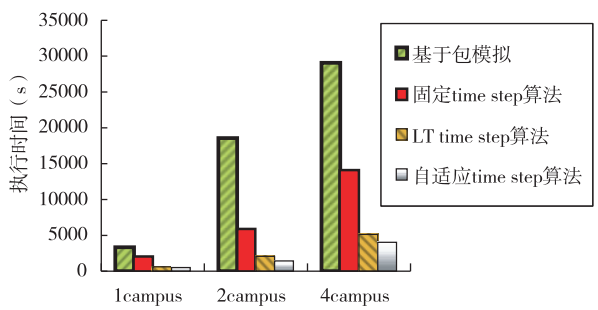


图 8　不同网络规模情况下四种模拟算法的执行时间

5　结　论

本文通过对网络模拟流量模型的分析,深入研究了基于时间驱动流量模型的流模拟方式,并将自适应龙格库塔算法与该模型合理结合,针对当前 time step 选取算法中存在的问题,设计了一种自适应 time step 选取算法,实现了网络模拟的高效性和高真实性。通过对自适应 time step 算法与其它 time step 算法在精确度、误差和效率上的实验分析,结果表明在保证流量模型真实性的前提下,本文提出的方案可根据网络流量变化的复杂程度实时设定 time step 并不断更新网络模型相关参数的值,实现了提高网络模拟精确度的同时,又提高了计算效率。实验也表明该方案对于大规模网络模拟环境也具有一定的通用性。因此,本文提出的针对基于时间驱动流量模型的自适应 time step 选取算法为下一步研究与建立大规模高真实性的实时网络模拟平台提供了理论依据和技术支持。

参考文献

[1] Ammar M H. Why we still don't know how to simulate networks. In Proceedings of the 13th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Atlanta, USA, 2005. 179-182

[2] 雷擎,王行刚. 计算机网络模拟方法与工具. 通信学报, 2001,22(9):84-90

[3] 王晓峰,提高大规模离散事件网络模拟性能方法的研究:[博士学位论文]. 哈尔滨:哈尔滨工业大学,2007.10

[4] Nicol D, Yan G. Discrete event fluid modeling of background TCP traffic. *ACM Transactions on Modeling and Computer Simulation*, 2004, 14:1-39

[5] Cole R, Riley G, Cansever D, et al. Stochastic process models for packet/analytic-based network simulations. In: Proceedings of the Workshop on Principles of Advanced and Distributed Simulation, 2008

[6] Misra V, Gong W B, Towsley D. Fluid-based analysis of a network of AQM routers supporting TCP flows with an ap-

plication to RED, In: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, 2000, 151-160

[7] Liu Y, Presti F, Misra V, et al. Scalable fluid models and simulations for large-scale IP networks. *ACM Transactions on Modeling and Computer Simulation*, 2004, 14(3): 305-324

[8] Kumar S, Park S, Iyengar S, et al. Time adaptive numerical simulation for high speed networks. In: Proceedings of the International Conference on High Performance Computing, Networking and Communication Systems, 2007. 198-205

[9] Li T, Liu J. A fluid background traffic model. In: Proceedings of the IEEE International Conference on Communications, 2009

[10] Liu J. Packet-level integration of fluid TCP models in real-time network simulation, In: Proceedings of the 2006 Winter Simulation Conference, 2006

[11] 姜健飞等. 数值分析及其 MATLAB 实验. 北京: 科学出版社, 2004. 136-154

[12] 徐安农. 科学计算引论: 基于 mathematica 的数值分析. 北京: 机械工业出版社, 2010. 232-241

[13] 施妙根, 顾丽珍. 科学和工程计算基础. 北京: 清华大学出版社, 1999. 204-217

[14] Press W H, Flanney B P, Teukolsky S A, et al. Numerical Recipes-The Art of Scientific Computing. England: Cambridge University Press, 1992. 707-723

[15] Liu J, http://www.cis.fiu.edu/~liux/research/projects/prime/, 2007

Research on the network simulation fluid model based on an adaptive algorithm for time step selection

Sun Yue[*,**], Hao Zhiyu[*], Yun Xiaochun[*], Fei Haiqiang[*]

([*] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093)

([**] School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876)

Abstract

The network fluid simulation of time-driven fluid models was deeply studied through the analysis of present fluid models for network simulation, and considering the deficiencies of current time-driven fluid models in time step selection, a new algorithm for adaptive selection of time step was proposed which adjusts the time step continuously according to the complexity of network behaviors. The experimental results show that the adaptive time step selection algorithm can reduce the execution time for network simulation obviously while increasing the simulation accuracy compared with the other strategies. Furthermore, good effects can also be obtained for large-scale networks when using the adaptive time step selection algorithm, which will provide the foundation for studying the large-scale real-time network simulations.

Key words: network simulation, fluid model, adaptive Runge-Kutta, PRIME platform, performance optimization