

GPU OpenFlow 海量数据网络处理模型——GOMDI

张 伟^{1*}, 解争龙¹, 丁要军^{1,2}, 张潇晓²

(1. 咸阳师范学院 信息工程学院, 陕西 咸阳 712000; 2. 西北工业大学 计算机学院, 西安 710072)

(* 通信作者电子邮箱 mrweizhang@163.com)

摘 要: OpenFlow 的出现提高了现有网络的服务质量(QoS),但在处理海量数据时存在网络会话识别效率低、网络报文转发路径不佳等缺点。在 OpenFlow 的研究基础上,提出了海量网络数据处理(GOMDI)模型,通过将 GPU 并行计算、生物序列算法和机器学习方法相融合,设计出 GOMDI 网络会话匹配算法和路径选择算法。实验结果表明, GOMDI 网络会话匹配算法与 CPU 环境相比加速比提升了近 300;路径选择算法中网络丢包率低于 5%,网络延时小于 20 ms。因此, GOMDI 模型可有效地提升网络性能,满足大数据环境下实时处理海量信息的需求。

关键词: OpenFlow; GPU; 生物序列; 机器学习

中图分类号: TP393.02 **文献标志码:** A

GOMDI: GPU OpenFlow massive data network analysis model

ZHANG Wei^{1*}, XIE Zhenglong¹, DING Yaojun^{1,2}, ZHANG Xiaoxiao²

(1. College of Information Engineering, Xianyang Normal University, Xianyang Shaanxi 712000, China;

2. College of Computer, Northwestern Polytechnical University, Xi'an Shaanxi 710072, China)

Abstract: OpenFlow enhances the Quality of Service (QoS) of traditional networks, but it has disadvantage that its network session identification efficiency is low and the network packet forwarding path is poor and so on. On the basis of the current study of the OpenFlow, GPU OpenFlow Massive Data Network Analysis (GOMDI) model was proposed by this paper, through integrating the biological sequence algorithm, GPU parallel computing algorithm and machine learning methods. The network session matching algorithm and path selection algorithm of GOMDI were designed. The experimental results show that the speedup of the GOMDI network session matching algorithm is over 300 higher than the CPU environment in real network, and the network packet loss rate of its path selection algorithm is lower than 5%, the network delay is less than 20 ms. Thus, the GOMDI model can effectively improve network performance and meet the needs of the real-time processing for massive information in big data environment.

Key words: OpenFlow; Graphic Processing Unit (GPU); biological sequence; machine learning

0 引言

随着网络技术的发展,创新对现有网络更加困难。为了更好地满足下一代网络的需求,Greenberg 等^[1]和 McKeown 等^[2]提出了支持网络会话管理的新型网络 OpenFlow,弥补了现有网络的不足。网络会话的管理,能够提供更高的服务质量,更多的服务类型。OpenFlow 是基于交换机安装,每台交换机上有网络会话表和增删网络会话表项,能有效判别网络会话类型,依据规则对会话进行处理。如何快速、高效地识别网络会话,高质量转发会话是目前遇到的问题。针对这些,Moore 等^[3]提出了基于载荷的识别方法,以克服固定端口识别的不足,该方法对资源的需求较高,依赖于扫描载荷,不利于隐私保护。快速定位的方法有两种:一种是采用优化算法,如生物序列的方法;另一种是利用专用的硬件设备,而专门硬件成本太高,不利于普及,使用 GPU(Graphic Processing Unit)计算是目前替代方法。会话转发依据 IP 地址和网络状态,

选取符合要求的转发路径。以往根据转发表进行,通过匹配转发表中信息,选择合适路径,但转发表中路径不都是最优。OpenFlow 提供了更多服务功能,如何合理、高效地利用这些功能,提升转发质量,是亟待解决的问题。当前可采取的方法有利用机器学习的方法,如多层感知器、支持向量机和神经网络等,对历史信息训练,寻找最优路径。

本文针对目前 OpenFlow 交换机如何快速、高效识别会话,高质量转发的问题,提出将 GPU 并行计算与生物序列的方法结合,提升会话识别速度和准确度,同时结合机器学习的方法,选择合适的转发路径。

1 相关工作

1.1 生物序列的方法

生物序列的方法主要用于对蛋白质中氨基酸的相似性和同源性的匹配,主流的方法有 Needleman 等^[4]于 1970 年提出的 NW(Needleman-Wunsch)算法、Smith 等^[5]于 1981 年提出

收稿日期: 2014-03-11; **修回日期:** 2014-04-18。 **基金项目:** 国家自然科学基金资助项目(61102018);陕西省教育厅科研计划项目(12JK0933);陕西省科技厅科研计划项目(2013JM8037);咸阳师范学院专项科研基金资助项目(12XSYK068)。

作者简介: 张伟(1981-),男,陕西咸阳人,讲师,硕士,主要研究方向:并行计算、机器学习; 解争龙(1961-),男,陕西咸阳人,教授,硕士,主要研究方向:网络安全; 丁要军(1980-),男(回族),河南许昌人,讲师,博士,主要研究方向:机器学习、网络安全; 张潇晓(1986-),男,湖南株洲人,博士研究生,主要研究方向:云计算、金融分析、机器学习。

了SW(Smith-Waterman)算法,Aji等^[6]于2008年提出的WF(WaveFront)算法和Martins等^[7]于2001年提出的TWF(Tiled WaveFront)算法,这些算法在解决相似性快速比对方面产生了巨大影响,然而由于算法自身的特点,存在些许的不足。本节主要介绍这些算法的优缺点,作为OpenFlow网络会话匹配的理论依据。

NW算法最初被应用于解决蛋白质中氨基酸序列相似性,判断是否为同源蛋白质。该算法首先初始化矩阵用于全局比对,初始化方式如式(2)所示,纵横轴序列分别代表需要比对的序列。利用递归的方式访问 H_{ij} 并计算其数,访问计算如式(1)所示。其中: $S(a_i, b_j)$ 为比对序列 a_i 和 b_j 的相似性, w_k 为对空隙的惩罚。通过计算可以得到两序列的最高分,再通过回溯矩阵得到两比对序列的匹配结果,该算法适合对长度和内容都很相似的序列进行比较,不适合对序列片的比较。

$$H_{ij} = \max \left\{ H_{i-1, j-1} + S(a_i, b_j), \max_{1 \leq k \leq i} (H_{i-k, j-1} - w_k + S(a_i, b_j)), \max_{1 \leq k \leq j} (H_{i-1, j-k} - w_k + S(a_i, b_j)) \right\} \quad (1)$$

其中:

$$H_{00} = H_{0j} = 0 \quad (2)$$

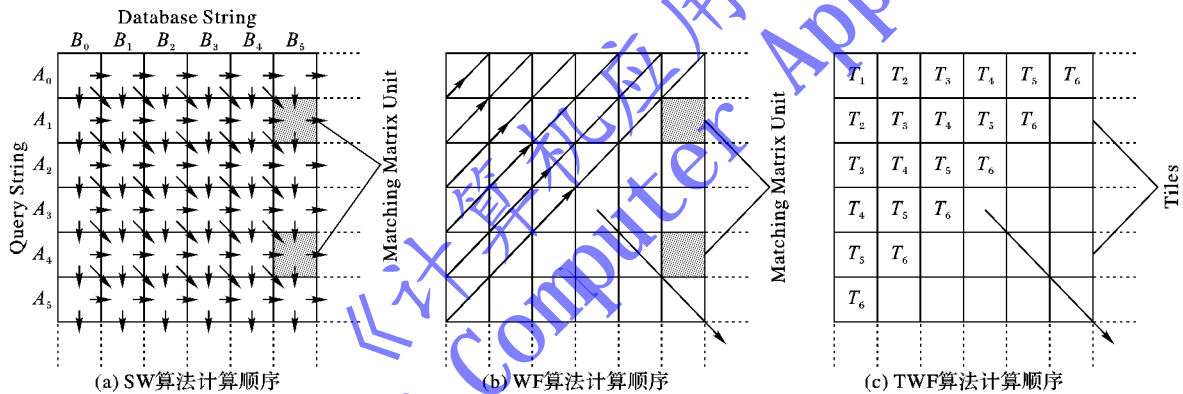


图1 SW、WF和TWF算法计算顺序

TWF算法可以有效解决文献[6]算法的通信开销问题。Martins等^[7]实现了将多线程并行计算与TWF算法的融合。TWF在文献[6]算法上将多个矩阵归并为一块,以降低矩阵间通信开销对整体性能的影响。块合并即把 $R \times C$ 矩阵元素作为完整块对待,实现全部块的计算需复制块左方 R 个元素,上方 C 个元素和左上方1个元素,共 $(R+C+1)$ 个数据元素。块由 $R \times C$ 个矩阵元素构成,每次循环迭代计算反对角线上一排块,TWF算法如图1(c)所示。

综上所述,生物序列方法能够很好解决序列的比对问题,这对于具有海量数据OpenFlow交换机而言尤为重要,但是如何更加高效、高准确度地利用生物序列的方法对流经OpenFlow交换机的网络数据报文进行匹配显得尤为重要。

1.2 机器学习的方法

机器学习是通过对已有知识的学习,为以后的实践提供依据的过程,Simon^[8]认为学习是自适应变化的过程,使得系统能够更有效地执行相同或相似的任务。文献[9-10]以为学习是组织或修正关于所经历事物的表现。专家系统研制的人认为学习是获得知识的过程。综上所述,第1种观点认为学习是外部的行为;第2种观点认为学习是内在的过程;而第

3种观点认为学习从实际的知识工程出发。机器学习的方法有很多,主要有支持向量机(Support Vector Machine, SVM)、人工神经网络(Artificial Neural Network, ANN)、模糊系统(Fuzzy System)等,结合本文的研究重点,是对已有的转发规则和会话进行学习,将学习后的模式应用于网络会话转发,从而提供更好、更优质的网络服务。而转发的网络会话数据具有海量、高并发、相似性高等特点,针对这些特点,目前可采取的方法有SVM^[11]、K-means等。本节主要介绍这些算法的相关研究的适应性和优缺点,为OpenFlow网络会话转发的提供理论依据。

$$H_{ij} = \max \left\{ H_{i-1, j-1} + S(a_i, b_j), \max_{1 \leq k \leq i} (H_{i-k, j-1} - w_k), \max_{1 \leq k \leq j} (H_{i-1, j-k} - w_k), 0 \right\} \quad (3)$$

其中

$$H_{00} = H_{0j} = 0 \quad (4)$$

WF算法改变了计算方式,其思想是从沿 H_{ij} 对角线方向的左上角开始迭代,计算反对角线所有矩阵元素,如图1(b)所示,因为矩阵反对角线方向元素具备独立性,所以可采取并行方式计算。而如果利用文献[5]算法会造成大批的通信,这是由于每计算一个元素需复制上方、左方和左上方3个单位的数值。

3种观点认为学习从实际的知识工程出发。机器学习的方法有很多,主要有支持向量机(Support Vector Machine, SVM)、人工神经网络(Artificial Neural Network, ANN)、模糊系统(Fuzzy System)等,结合本文的研究重点,是对已有的转发规则和会话进行学习,将学习后的模式应用于网络会话转发,从而提供更好、更优质的网络服务。而转发的网络会话数据具有海量、高并发、相似性高等特点,针对这些特点,目前可采取的方法有SVM^[11]、K-means等。本节主要介绍这些算法的相关研究的适应性和优缺点,为OpenFlow网络会话转发的提供理论依据。

Cortes等^[11]利用SVM分析小样本、非线性和高维模式识别。Este等^[12]提出了使用SVM对网络数据流分类的方法,该方法虽然对样本的规模要求较小,但依赖于样本的准确度。Li等^[13]针对训练样本的先验概率致使严重偏离的问题,提出利用SVM的方法,训练7类不同特点的应用,提取应用特征,找到最好组合,其分类准确度达99.4%,该方法需要对样本进行筛选。尽管以上的研究分类精度和样本挑选,提出了创新的方法,但难以在大规模训练样本中实施,主要原因是SVM需要二次规划来求解支持向量,而二次规划需计算 n 阶矩阵,其时间复杂度和空间复杂度很高。Platt^[14]提出SMO

(Sequential Minimal Optimization)算法,Joachims^[15]提出 SVM, Burges^[16]提出 PCGC (Parallel Concatenated Gallager Code) 和 Mangasarian 等^[17]提出 SOR (Successive Over Relaxation) 算法用于改善二次规划中 n 阶矩阵的计算。

K -means 方法^[18]在小于 10000 数据方面相对有优势,Liu 等^[19]提出使用无监督的 K -means 对网络流量进行分类,该方法可获得高达 80% 的整体精度,该方法虽然分类准确度取得了不错的效果,但是对于那些以动态端口的协议而言,分类效果显得不足。Erman 等^[20]针对 P2P 使用动态端口号、伪装技术和加密,使得检测越来越困难,提出了使用 K -means 方法和 DBSCAN (Density-Based Spatial Clustering of Applications with Noise) 方法利用应用程序的特点对流量分类,该方法表明 K -means 准确精度高而 DBSCAN 产生更好聚类。

综上所述,机器学习的方法对于样本的学习,现实的工作提供依据十分重要,但是无论是支持向量机还是 K -means 都存在或多或少的问题,问题主要围绕在样本的预选和样本规模两方面,支持向量机对样本的精确度要求高,以及对样本的

规模有限制,对于大规模的样本,训练的开销大。而 K -means 方法对多于 10000 的样本,训练开销过大,训练结果不佳。

2 本文海量网络流量处理模型

GPU OpenFlow massive network data analysis (GOMDI) 模型由两部分组成,分别是网络会话匹配和网络会话转发:网络会话匹配使用生物序列方法算法,该算法基于 GPU 框架下实现;网络会话转发使用的是会话等级排序算法。以下主要介绍 GOMDI OpenFlow Switch 的模型、网络会话匹配算法和网络会话转发算法。

2.1 GOMDI OpenFlow Switch 海量网络流量处理模型的架构

GOMDI OpenFlow Switch 将 GPU 有效融合进 OpenFlow Switch 中,用于提升网络匹配的准确度和效率,网络报文从网口进来,由 Flow Table 模块将需要进行匹配的网络会话导入 GPU Pattern Match,由该部分完成对网络会话的规则匹配,并对网络会话进行标识。GOMDI OpenFlow Switch 处理框架如图 2 所示。

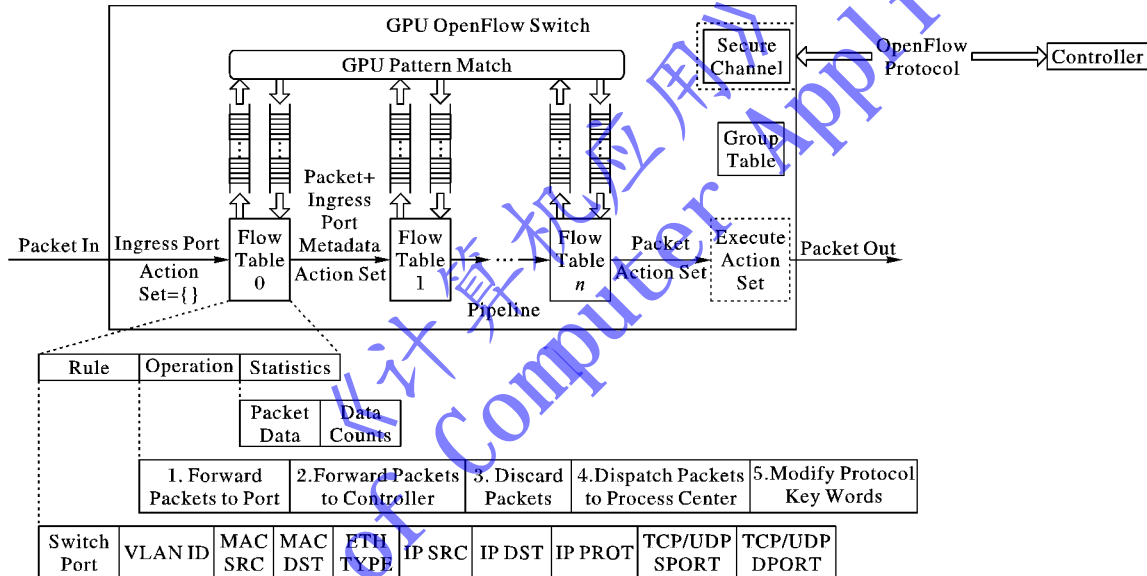


图2 OpenFlow Switch 总体架构

2.2 GOMDI OpenFlow Switch 网络会话匹配算法

GOMDI OpenFlow Switch 网络会话匹配算法在 GPU 的基础上实现对网络会话的高效匹配,实现对网络会话的线速处理,本节主要介绍 GPU 的相关概念和网络会话匹配算法。

2.2.1 相关概念

1) Kernel。指的是 CUDA GPU 中的运行函数。

2) 线程块。指的是线程的集合,线程块中的线程相互独立,运行顺序不依赖于同一线程块中其他线程。本文假设每个线程块中有 N 个线程。

3) SM。在硬件中调度并发线程,调度不存在额外的开销,线程在 SM 上运行,每个 SM 上能够运行多线程块。

2.2.2 算法分析

第 1.1 节介绍了主要的生物序列匹配算法,虽然 TWF 算法的通信开销相比于文献[6]算法有所减少,但仍未充分利用 GPU 的并行处理能力。主要是每块仅有一线程参与计算,且效率低、稳定性差、可靠性不高,为提升线程对系统资源的利用率,将 BLAST 算法应用于 GPU 环境中(称为 GPU BLAST

算法),克服 TWF 算法的不足。在 BLAST 算法集中,起到核心作用的是 SCAN 函数,因此本文主要对 SCAN 并行化。

SCAN 首先首先遍历网络会话库 (SessionAccess), 对每 Block 块与待查序列 (BlockQuery) 进行匹配,命中时 (Hit) 时,把 SessionAccess 和 BlockQuery 的位置偏移信息 (Offset Information) 添加至记录中,SCAN 函数数据读取模式如图 3 所示。

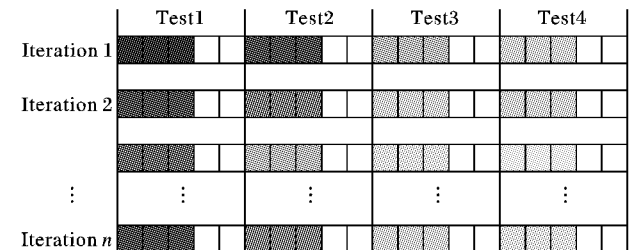


图3 SCAN 函数数据读取模式

SCAN 以 84 B 偏移量为步长单位,遍历 SessionAccess,每次迭代处理 21 B,迭代由 4 步组成,分别是 Test1、Test2、Test3

和 Test4。每步中,SCAN 在 SessionAccess 中读取 2 B 或 3 B 进行哈希运算,对生成的关键值进行匹配。

2.2.3 算法实现

SCAN 并行性方式有两种:

- 1) 循环并行,每次循环为一独立任务。
- 2) Test 并行,每步为一独立任务。

本文选用第 2) 种方式,主要因为 Test 之间并行化,可利用更多的线程,并行化加速效果明显。Test 并行访问方式,如图 4 所示。

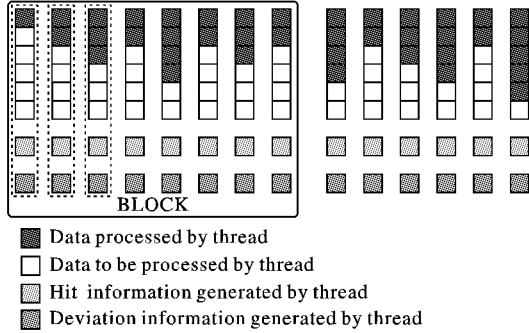


图 4 Test 并行访问方式

CUDA 并行化程序的主要参数:

- 1) 线程数目(Thread)。由 2.2.2 节可知,每步 Test 块大小为 21 B,因此 SCAN 线程数 $Thread = Length(SessionAccess)/21$ 。
- 2) 线程块数目(Block)。线程块中有 256 线程,因此 $Block = Length(SessionAccess)/(21 \times 256)$ 。

2.3 GOMDI OpenFlow Switch 网络会话转发算法

2.3.1 主要思想和相关概念

在 GOMDI OpenFlow Switch 网络会话转发算法中,首先需对相关的网络路径进行检索,检索返回两种值:1) 权威值,指所有链入路径的网络会话中心值之和;2) 中心值,指网络会话中所有链出路径指向网络会话的权威值之和。算法执行中迭代的基本步骤:权威值更新和中心值更新,网络路径中节点的中心值和权威值计算方法如算法 1 所示。

- 1) 权威值更新规则。

$\forall p$,更新权威值 $auth(p)$,如式(5)所示:

$$auth(p) = \sum_{i=1}^n hub(i) \quad (5)$$

- 2) 中心值更新规则。

$\forall p$,更新中心值 $hub(p)$,如式(6)所示:

$$hub(p) = \sum_{i=1}^n auth(i) \quad (6)$$

其中: n 表示网络会话数, i 是网络会话 p 所选择的传输路径。

算法 1 Node auth and hub value calculate.

输入 Node。

输出 $auth(p)$, $hub(p)$ 。

- 1) Init $\forall p, auth(p) = 1 \& hub(p) = 1$ //初始化中心值和权威值
- 2) $auth_update_rule(p)$ //更新权威值规则
- 3) $hub_update_rule(p)$ //更新中心值规则
- 4) repeat Step 2)

2.3.2 GOMDI OpenFlow Switch 网络会话转发算法

在开始执行时, $\forall p, auth(p) = 1$ 和 $hub(p) = 1$,假设两种类型更新规则:权威值更新规则和中心值更新规则。为了计算每个节点的权威值和中心值,需重复迭代权威值更新规则

和中心值更新规则。权威值和中心值的更新相互交替,相互影响,算法的具体步骤如算法 2 所示。

算法 2 GOMDI OpenFlow Switch Network Session Calculate Algorithm。

输入 PacketSessions。

输出 $auth$, hub values of PacketSessions。

$G \leftarrow PacketSessionSet$

InitPacketSession(G)

//初始化 G

Hub_Auth_Update(G)

//更新算法

for 1 to k

//运行算法 k 步

$norm \leftarrow 0$

for $\forall PacketSession p \in G$

//更新所有的权威值

$p.auth \leftarrow 0$

PacketSession_In(PacketSession p)

// $p.in$ 指向 G

$norm += square(p.auth)$

//计算总的权威值平方和,并归一化

end for

$norm = sqrt(norm)$

end for

for $\forall PacketSession p \in G$

//更新权威值

$p.auth \leftarrow p.auth/norm$

//权威值归一化

end for

$norm \leftarrow 0$

for $\forall PacketSession p \in G$

//更新所有的中心值

$p.hub \leftarrow 0$

for PacketSession_Out(PacketSession p)

// $p.out$ 指向 G

$norm += square(p.hub)$

//计算总的中心值平方和,并归一化

end for

$norm = sqrt(norm)$

end for

for $\forall PacketSession p \in G$

//更新所有的中心值

$p.auth \leftarrow p.auth/norm$

//中心值归一化

end for

end Hub_Auth_Update(G)

3 GOMDI OpenFlow Switch 的实验测试

GOMDI OpenFlow Switch 的实验测试采用服务器配备有 8 个 Xeon E7-8830 处理器,内存 128 GB,两块 Nvidia Tesla K40 显卡,每块拥有 12 GB 显存和 2 880 颗 CUDA 核,网卡选用两块双光纤端口 10 000 Mb/s 网卡,传输速率峰值可达 40 000 Mb/s,安装 Cent OS 64 bit 操作系统,数据集采集至骨干链路网实时数据,数据大小达 1 247 GB,流表容量达 370 254 条。实验主要测试 GOMDI OpenFlow Switch 两方面的性能:1) GOMDI OpenFlow Switch 网络会话匹配效率。该方面主要对比测试 GPTWF 算法分别在 CPU 和 GPU 环境下的加速比,加速比测试选用的流表容量从 50 000 条至 250 000 条,数据大小从 10 GB 至 100 GB。2) GOMDI OpenFlow Switch 转发算法的转发效果。该方面的测试主要围绕转发链路状态进行,测试选择的转发路径是否为最佳路径,路径是否存在拥塞等,主要通过数据集网络会话报文的延时等信息进行分析。

3.1 GOMDI OpenFlow Switch 网络会话匹配算法对比测试

GOMDI OpenFlow Switch 网络会话匹配算法的测试分别测试 Needleman-Wunsch、WaveFront、Smith-Waterman、TWF 和 GPU BLAST 算法在 CPU 和 GPU 下的运行时间及加速比,CPU 和 GPU 运行时间如表 1 所示,其中: NW^* 为 Needleman-

Wunsch; WF* 为 WaveFront; SW* 为 Smith-Waterman; GP* 为 GPU BLAST。实验测试表明, GPU BLAST 算法在海量数据下

的加速比峰值接近 300, 远优于其他算法, 能够很好地实现海量数据下的网络会话匹配。

表 1 生物序列网络会话匹配算法在 CPU、GPU 的运行时间对比

s

处理器	Open-Flow 数据容量/GB	OpenFlow Flow Table 容量																	
		50 000						150 000						250 000					
		CPU	NW*	WF*	SW*	TWF	GP*	CPU	NW*	WF*	SW*	TWF	GP*	CPU	NW*	WF*	SW*	TWF	GP*
CPU	10	3.572	2.659	2.333	1.868	1.395	0.886	13.76	8.999	8.01	6.789	5.458	3.834	19.57	13.07	12.72	11.61	7.889	4.525
	40	11.740	11.960	10.270	8.565	7.452	4.207	46.63	37.650	35.30	29.470	26.410	17.230	106.90	57.24	50.37	40.08	39.120	26.600
	70	24.360	24.890	19.650	16.450	13.210	8.729	82.27	78.600	69.60	58.690	47.540	38.030	167.60	132.70	114.00	107.40	92.450	65.020
	100	36.540	32.500	29.420	24.740	21.650	15.840	156.30	114.000	90.67	76.750	69.450	46.270	245.10	190.90	174.00	150.40	128.000	96.040
GPU	10	3.627	0.109	0.045	0.051	0.029	0.007	12.81	0.299	0.301	0.179	0.116	0.032	19.59	0.418	0.472	0.272	0.142	0.059
	40	11.540	0.496	0.290	0.273	0.149	0.065	45.85	1.762	1.407	0.821	0.574	0.209	106.90	2.690	1.589	1.356	0.720	0.305
	70	24.030	1.091	0.527	0.503	0.300	0.124	81.71	3.874	2.726	1.734	1.105	0.386	167.20	6.538	3.719	3.168	2.121	0.792
	100	36.360	1.169	1.028	0.678	0.491	0.196	155.40	5.446	3.218	2.295	1.439	0.649	238.40	8.006	5.658	4.438	2.770	1.265

3.2 GOMDI OpenFlow Switch 网络会话转发算法对比测试

本节主要测试采用算法后网络链路的状态。测试采用事后测试的方式进行,即在运行转发算法一段时间后网络状态怎样,同时对比不采用 GOMDI OpenFlow 转发算法的网络链路状态,测试同一链路下的丢包率和延时,测试链路带宽为 10 Gb/s。网络丢包测试对比如图 5(a)所示,网络延时对比如图 5(b)所示。实验表明使用转发算法丢包率和延时整体呈下降趋势,分别平均下降 74.64% 和 68.03%,其计算如(7)所示:

$$AverageRate = \sum_{i=1}^n \frac{y_i^{No} - y_i^{Yes}}{i \times y_i^{No}} \times 100\% \quad (7)$$

其中: i 表示总的统计数目, y_i^{No} 表示未采用算法的数据, y_i^{Yes} 表示采用算法的数据。

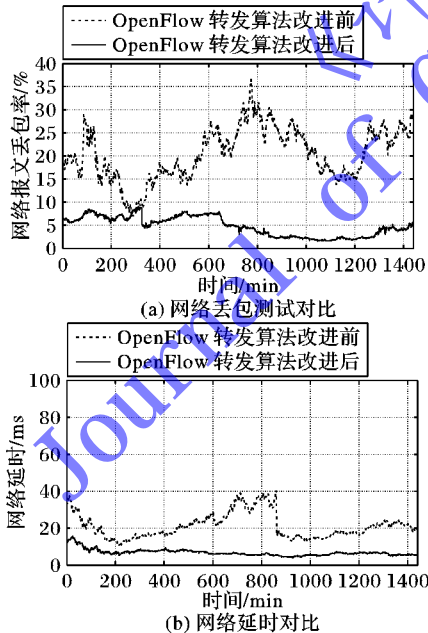


图 5 OpenFlow 网络会话转发算法对比测试

4 结语

本文把生物序列对比算法和 GPU 进行结合应用于 OpenFlow 的网络会话匹配,并采用机器学习的方法,对网络会话的属性进行学习,将网络会话转发到合适的网络链路。本文提出了 GOMDI 模型,该模型能够很好地用于海量网络流

量的处理,在网络会话匹配上,将生物序列的匹配算法和 GPU 相结合,获得了近 300 的加速比;通过采用机器学习的方法,对网络会话进行分析,将网络会话转发至合适的网络链路,优化网络环境,实验结果表明转发算法使得链路丢包率低于 5%,平均下降 74.64%;延时小于 20 ms,平均下降 68.03%,在网络丢包率和延时方面能取得不错的效果。本文的下一步工作将更进一步优化处理模型。

参考文献:

- [1] GREENBERG A, HJALMTYSSON G, MALTZ D A, *et al.* A clean slate 4D approach to network control and management [J]. ACM SIGCOMM Computer Communication Review, 2005, 35(5): 41 - 54.
- [2] McKEOWN N, ANDERSON T, BALAKRISHNAN H, *et al.* OpenFlow: enabling innovation in campus networks [J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69 - 74.
- [3] MOORE A W, PAPAGIANNAKI K. Toward the accurate identification of network applications [C]// Proceedings of the 6th International Conference on Passive and Active Network Measurement. Berlin: Springer, 2005: 41 - 54.
- [4] NEEDLEMAN S B, WUNSCH C D. A general method applicable to the search for similarities in the amino acid sequence of two proteins [J]. Journal of Molecular Biology, 1970, 48(3): 443 - 453.
- [5] SMITH T F, WATERMAN M S. Identification of common molecular subsequences [J]. Journal of Molecular Biology, 1981, 147(1): 195 - 197.
- [6] AJI A M, FENG W, BLAGOJEVIC F, *et al.* Cell-SWat: modeling and scheduling wavefront computations on the cell broadband engine [C]// Proceedings of the 5th Conference on Computing Frontiers. New York: ACM Press, 2008: 13 - 22.
- [7] MARTINS W S, del CUVILLO J, USECHE F J, *et al.* A multithreaded parallel implementation of a dynamic programming algorithm for sequence comparison [C]// PSB 2001: Proceedings of the 6th Pacific Symposium on Biocomputing. Hawaii: [s. n.], 2001. 2001: 311 - 322.
- [8] SIMON H A. The sciences of the artificial [M]. Cambridge: MIT Press, 1996.
- [9] MICHALSKI R S. A theory and methodology of inductive learning [J]. Artificial Intelligence, 1983, 20(2): 111 - 161.
- [10] ANDERSON J R. Machine learning: an artificial intelligence approach [M]. San Francisco: Morgan Kaufmann Publishers, 1986.

(下转第 2272 页)

- ceneters-waste-vast-amounts-of-energy-belying-industry-image. html.
- [5] U. S. Environmental Protection Agency. EPA report on server and data center energy efficiency [EB/OL]. [2014-01-18]. http://www.energystar.gov/index.cfm?c=prod_development.server_efficiency_study.
 - [6] BATTLES B, BELLEVILLE C, GRABAU S, *et al.* Reducing data center power consumption through efficient storage [EB/OL]. [2014-01-05]. <http://bnrg.eecs.berkeley.edu/~randy/Courses/CS294.F07/NetApp2.pdf>.
 - [7] WANG X, WANG Y. Coordinating power control and performance management for virtualized server cluster [J]. *IEEE Transactions on Parallel and Distributed Systems*, 2011, 22(2): 245–259.
 - [8] SONG Y, WANG H, LI Y, *et al.* Multi-tiered on-demand resource scheduling for VM-based data center [C]// CCGRID 2009: Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. Washington, DC: IEEE Computer Society: 148–155.
 - [9] LIAO B, YU J, ZHANG T, *et al.* Energy-efficient algorithms for distributed file system HDFS [J]. *Chinese Journal of Computers*, 2013, 36(5): 1047–1064. (廖彬, 于炯, 张陶, 等. 基于分布式文件系统 HDFS 的节能算法[J]. *计算机学报*, 2013, 36(5): 1047–1064.)
 - [10] LIAO B, YU J, SUN H, *et al.* Energy-efficient algorithms for distributed storage system based on data storage structure reconfiguration [J]. *Journal of Computer Research and Development*, 2013, 50(1): 3–18. (廖彬, 于炯, 孙华, 等. 基于存储结构重配置的分布式存储系统节能算法[J]. *计算机研究与发展*, 2013, 50(1): 3–18.)
 - [11] PINHEIRO E, BIANCHINI R, DUBNICKI C. Exploiting redundancy to conserve energy in storage systems [C]// Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems. New York: ACM Press, 2006: 15–26.
 - [12] COLARELLI D, GRUNWALD D. Massive arrays of idle disks for storage archives [C]// Proceedings of the 2002 ACM/IEEE Conference on Supercomputing. Washington, DC: IEEE Computer Society, 2002: 1–11.
 - [13] HARNIK D, NAOR D, SEGALL I. Low power mode in cloud storage systems [C]// IPDPS 2009: Proceedings of the 2009 IEEE International Symposium on Parallel and Distributed Processing. Piscataway: IEEE Press, 2009: 1–8.
 - [14] LEVERICH J, KOZYRAKIS C. On the energy (in) efficiency of Hadoop clusters [J]. *ACM SIGOPS Operating Systems Review*. 2010, 44(1): 61–65.
 - [15] NITESH M, NANDURI R, VARMA V. Dynamic energy efficient data placement and cluster reconfiguration algorithm for MapReduce framework [J]. *Future Generation Computer Systems*, 2011, 28(1): 119–127.
 - [16] NARAYANAN D, DONNELLY A, ROWSTRON A. Write off-loading: practical power management for enterprise storage [J]. *ACM Transactions on Storage*, 2008, 4(3): 253–267.
 - [17] HAMILTON J. Cooperative Expendable Micro-Slice Servers (CEMS): low cost, low power servers for Internet-scale services [EB/OL]. [2014-01-11]. http://www.mvdrna.com/jrh/talk-sandpapers/jameshamilton_cems.pdf.
 - [18] VASUDEVAN V, FRANKLIN J, ANDERSEN D, *et al.* FAWN-damentally power-efficient clusters [C]// Proceedings of the 12th Conference on Hot Topics in Operating Systems. Berkeley: USENIX Association, 2009: 22.
 - [19] SZALAY A S, BELL G, HUANGZ H H, *et al.* Low-power AMDahl-balanced blades for data intensive computing [J]. *ACM SIGOPS Operating Systems Review*, 2009, 44(1): 71–75.
 - [20] LIM K, RANGANATHAN P, CHANG J, *et al.* Understanding and designing new server architectures for emerging warehouse-computing environments [C]// Proceedings of the 35th Annual International Symposium on Computer Architecture. Washington, DC: IEEE Computer Society, 2008: 315–326.
 - [21] McKNIGHT J, ASARO T, BABINEAU B. Digital archiving: end-user survey and market forecast 2006–2010 [EB/OL]. [2013-05-25]. <http://www.enterprisestrategygroup.com/ESGPublications/ReportDetail.asp?ReportID=591>.
 - [22] CALHEIROS R N, RANJAN R, BELOGLAZOV A, *et al.* Cloud-Sim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms [J]. *Software: Practice and Experience*, 2010, 41(1): 23–50.

(上接第2247页)

- [11] CORTES C, VAPNIK V. Support-vector networks [J]. *Machine Learning*, 1995, 20(3): 273–297.
- [12] ESTE A, GRINGOLI F, SALGARELLI L. Support vector machines for TCP traffic classification [J]. *Computer Networks*, 2009, 53(14): 2476–2490.
- [13] LI Z, YUAN R, GUAN X. Accurate classification of the Internet traffic based on the SVM method [C]// ICC'07: Proceedings of the 2007 IEEE International Conference on Communications. Washington, DC: IEEE Computer Society, 2007: 1373–1378.
- [14] PLATT J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods [M]// SMOLA A J, BARTLETT P J. *Advances in Large Margin Classifiers*. Cambridge: MIT Press, 2000: 61–74.
- [15] JOACHIMS T. Training linear SVMs in linear time [C]// Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2006: 217–226.
- [16] BURGESS C J C. A tutorial on support vector machines for pattern recognition [J]. *Data Mining and Knowledge Discovery*, 1998, 2(2): 121–167.
- [17] MANGASARIAN O L, STREET W N, WOLBERG W H. Breast cancer diagnosis and prognosis via linear programming [J]. *Operations Research*, 1995, 43(4): 570–577.
- [18] HARTIGAN J A, WONG M A. Algorithm AS 136: a k -means clustering algorithm [J]. *Journal of the Royal Statistical Society, Series C: Applied Statistics*, 1979, 28(1): 100–108.
- [19] LIU Y, LI W, LI Y. Network traffic classification using k -means clustering [C]// IMSCS 2007: Proceedings of the Second International Multi-Symposiums on Computer and Computational Sciences. Washington, DC: IEEE Computer Society, 2007: 360–365.
- [20] ERMAN J, ARLITT M, MAHANTI A. Traffic classification using clustering algorithms [C]// Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data. New York: ACM Press, 2006: 281–286.