NDN, Technical Report NDN-0005, 2012. http://named-data.net/techreports.html
Revision 2: October 5, 2012

ndnSIM: NS-3 NDN模拟器

Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang

f

命名数据网络(NDN)是一种新提出的互联网架构。NDN保留了互联网的沙漏架构，但发展了细腰。NDN不是将数据推送到特定位置，而是按名称检索数据。一方面，这种简单的改变允许NDN网络使用几乎所有经过良好测试的互联网工程属性，不仅解决IP的通信问题，而且解决数字分发和控制问题。另一方面，分布体系结构与当今互联网的点对点通信体系结构有着根本的不同，并提出了许多新的研究挑战。仿真可以作为一种灵活的工具来检查和评估这种新体系结构的各个方面。为了给研究界提供一个通用的仿真平台，我们开发了一个开源的基于NS-3的模拟器ndnSIM，它以模块化的方式忠实地实现了NDN网络的基本组件。本文对ndnSIM进行了概述。

1 简介

命名数据网络(NDN)[1]给互联网通信范式带来的根本变化要求对NDN设计的各个方面进行广泛和多维的评估。虽然现有的NDN实施(CCNx项目[2])，以及测试平台部署[3]，为评估NDN基础设施设计及其在现实环境中的应用提供了宝贵的机会，在大规模部署中，用不同的设计选项进行实验既困难，也不可能评估设计选择。为了满足这些需求，并为广大社区提供一个通用的实验平台，我们基于NS-3网络模拟器框架[4]开发了一个开源的NDN模拟器ndnSIM。

ndnSIM的设计有以下几个目标:

- 作为一个开源包，使研究界能够在一个通用的仿真平台上运行实验。
- 能够忠实地模拟所有基本的NDN协议操作。
- 与CCNx实现[2]保持包级互操作性，允许在CCNx和ndnSIM之间共享流量测量和数据包分析工具，以及直接使用真实的CCNx流量轨迹来驱动ndnSIM模拟实验。
- 能够支持大规模的仿真实验。
- 促进路由、数据缓存、数据包转发和拥塞管理的网络层实验。

遵循NDN架构，ndnSIM作为一种新的网络层协议模型实现，它可以运行在任何可用的链路层协议模型(点对点、CSMA、无线等)之上，也可以运行在网络层(IPv4、IPv6)和传输层(TCP、UDP)协议之上。这种灵活性允许ndnSIM模拟各种同构和异构部署场景(例如，NDN-only、NDN-over-IP等)。

模拟器以模块化的方式实现，使用单独的c++(一组)类来模拟NDN中每个网络层实体的行为:未决兴趣表(PIT)、转发信息库(FIB)、内容存储、网络和应用程序接口、兴趣反馈策略等。这种模块化结构允许任何组件都可以很容易地修改或替换，而对其他组件没有或只有很小的影响。此外，模拟器还提供了广泛的接口和助手集合，以执行每个组件和NDN流量的详细跟踪行为。

我们在2011年秋季开始了ndnSIM的实施工作。从那时起，最初的实现已经被我们自己用于各种NDN设计和评估任务，以及一些外部alpha测试人员。在我们继续积极开发ndnSIM的同时，ndnSIM作为开源包的第一个版本已于2012年6月发布。我们希望ndnSIM可以为对NDN研究感兴趣的更广泛的社区提供一个有用的工具，并且社区可以向我们提供宝贵的错误报告和新功能开发关于发行版、代码下载、基本示例和附加文档的更详细信息可在ndnSIM网站http://ndnsim.net/上获得。

2 design

创建开源NDN仿真包的愿望在很大程度上决定了我们选择NS-3网络模拟器[4]作为ndnSIM的基本框架。虽然NS-3是相对较新的，仍然没有

1. Bug reports and feature requests can be filed through GitHub social coding website interfaces https://github.com/NDN-Routing/ndnSIM.

商业Qualnet或遗留的ns-2模拟器所拥有的一切(例如，NS-3不支持模拟传统的动态IP路由协议2)，它提供了干净一致的设计，广泛的文档和实现灵活性。

在本节中，我们将提供有关ndnSIM设计的主要组件的见解，包括对协议实现组件的描述。

2.1 设计概述

ndnSIM的设计遵循了NS-3中网络仿真的理念，它为所有建模组件设计了最大的抽象。与现有的IPv4和IPv6栈类似，我们将ndnSIM设计为一个独立的协议栈，可以安装在模拟的网络节点上。除了核心协议栈之外，ndnSIM还包括一些基本的流量生成器应用程序和助手类，以简化模拟场景的创建(例如，在节点上安装NDN堆栈和应用程序的助手)和工具，以收集用于测量目的的模拟统计数据。

下面的列表总结了在ndnSIM中实现的组件级抽象;图1可视化了它们之间的基本交互:

- ndn::l3协议:实现核心NDN协议交互:通过面接收来自上层和下层的兴趣和数据包;
- ndn::Face:抽象实现与应用程序(ndn::AppFace)和其他模拟节点(ndn::NetDeviceFace)的统一通信，支持可插拔(和可选)的链路级拥塞缓解模块;
- ndn::内容存储:对数据包的网络内存储(例如，短期暂存，长期暂存，长期永久)的抽象;
- ndn::Pit:对未决兴趣表(PIT)的抽象，该表跟踪(每个前缀)收到兴趣的面，转发兴趣的面，以及以前看到的兴趣事件;
- ndn::Fib:转发信息库(FIB)的抽象，用于转发策略指导兴趣转发;
- ndn::ForwardingStrategy: Interest and Data转发的抽象和核心实现。转发过程中的每一步——包括查找内容存储、PIT、FIB和根据PIT项转发数据包——都被表示为虚拟函数调用，可以在特定的转发策略实现类中被覆盖(参见2.7节);
- 参考NDN应用，包括简单的流量生成器和接收器。

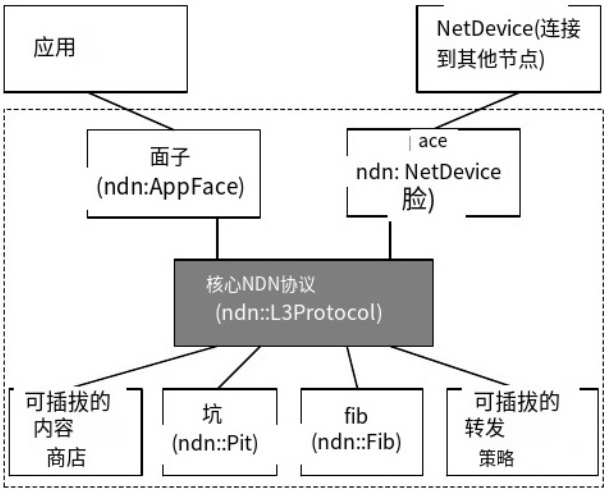

图1。ndnSIM组件框图

除了核心ndn:: l3协议之外，每个组件都有许多可选的实现，可以由模拟场景使用助手类任意选择(参见ndnSIM在线文档http://ndnsim.net/helpers.html).For示例)。ndnSIM目前为内容存储抽象提供了最近最少使用(LRU)、先进先出(FIFO)和缓存数据随机替换策略的实现。

当前兴趣和数据包的连线格式遵循现有CCNx项目的NDN实现(CCNx二进制XML编码3)的格式。虽然这种设计选择带来了额外的开销(即，二进制XML解码和编码过程有其成本)，但它允许重用现有的流量分析工具(ndndump，[4]wireshark ccnplugin[5])，以及使用真实的CCNx流量轨迹进行驱动模拟。稍后，我们将提供禁用数据包格式兼容性的选项，这将允许在运行大规模模拟时节省资源。

ndnSIM的设计包含多个可选模块，包括(1)数据安全占位符(当前代码允许在数据包上附加用户指定的“签名”)，(2)负确认的实验支持(Interest NACK)，提供数据平面问题的快速反馈。(3)可插拔的利率限制和接口可用性组件，以及(4)可扩展的统计模块。有兴趣的读者可以查看[5]了解更多关于利息NACKs和利率限制的详细信息。

2.2 核心NDN协议实现

ndnSIM中的ndn:: l3协议是一个中心体系结构实体，与

2. http://www.nsnam.org/docs/release/3.14/models/html/routing-overview.html

3. http://www.ccnx.org/releases/latest/doc/technical/BinaryEncoding.html

4. https://github.com/cawka/ndndump/

5. https://github.com/ProjectCCNx/ccnx/tree/master/apps/wireshark

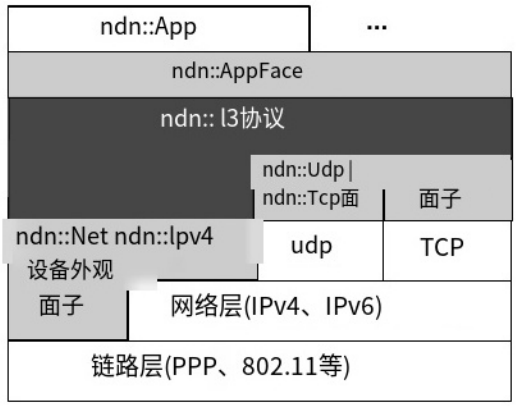

图2.。用于ndnSIM场景的通信层抽象

NS-3框架中相应的Ipv4L3Protocol和Ipv6L3Protocol类，实现IPv4和IPv6网络层协议。ndn:: l3协议是一个逻辑组件聚合器，用于与应用程序和其他节点之间的所有可用通信通道(面抽象，参见章节2.3)，并执行来自面的传入数据包的基本处理到转发策略。

l3协议类定义了API来操作NDN栈实现的以下方面:

- **AddFace/RemoveFace:** 注册一个新的人脸实现到NDN协议或删除现有的人脸;

2.3 人脸抽象

为了实现提供最大灵活性和可扩展性的目标，我们通过抽象层间交互使ndnSIM设计独立于底层传输。也就是说，核心协议实现(ndn::l3协议)、网络和应用之间的所有通信都是通过一个*Face抽象(ndn::Face)*来完成的，它可以有多种形式实现(见图2):链路层面(ndn::NetDeviceFace)用于节点间直接通过链路层通信，网络层面(ndn::Ipv4Face和ndn::Ipv6Face)和传输层面(ndn::TcpFace和ndn::UdpFace)用于节点间覆盖通信，应用层面(ndn::AppFace)用于节点内通信。

在第一次发布时，当前的ndnSIM包仅提供链路层ndn::NetDeviceFace和应用层ndn::AppFace。有了这两个面，就可以模拟一个完全支持ndn的网络。添加其他类型的面孔是很简单的，我们希望它们在模拟更复杂场景(即，NDN节点与不运行NDN协议的节点混合在一起)的需求出现时被添加，无论是我们自己还是社区中的其他人。

Face抽象定义了以下API:

- **SendImpl**(特定于实现):将数据包从NDN堆栈传递到底层(网络或应用程序)。
- RegisterProtocolHandler(特定于实现):允许从底层(网络或应用程序)转发数据包到NDN堆栈。
- SetMetric/GetMetric:分配和获取可以使用的Face metric，例如，在路由计算中。
- IsUp/SetUp:检查是否启用了Face，并启用/禁用Face。

除了基本API之外，Face抽象还提供了存储任意信息的方法，这些信息可以被转发策略模块使用。例如，一个ndn::fw::SimpleLimits兴趣转发策略基于物理链路限制[5]实现了对未完成兴趣数据包数量的限制。

2.4 内容库抽象

每个NDN路由器上的内容库支持网内存储，提供高效的错误恢复和异步多播数据传输。ndnSIM提供了一个接口，用于插入内容库的不同实现，这些实现可以实现不同的索引和项查找设计、不同的大小限制特性以及不同的缓存替换策略。

当前版本的ndnSIM包含了内容库抽象的三种实现，即最近最少使用(ndn::cs::Lru)、先进先出(ndn::cs::Fifo)和随机替换策略(ndn::cs::Random)。这些实现中的每一种都基于一个动态的基于尝试的容器，其大小有一个(可选的)上限，以及基于哈希的数据名称索引(在一个trie上按组件查找)。其他内容库模块可以由我们自己实现，也可以根据需要在社区的帮助下实现。

内容库抽象提供了以下操作:

- Add(特定于实现):在缓存中缓存新的或提升现有的数据包。
- 查找(特定于实现):对先前缓存的数据执行查找。

内容库抽象不提供显式的数据删除操作。内容库条目的生存期取决于流量模式，以及数据生产者提供的数据包新鲜度参数。

2.5 未决利息表(PIT)抽象

PIT (ndn:: PIT)维护每个转发的兴趣包的状态，为数据包转发提供方向。每个PIT条目包含以下信息:

- 与参赛作品相关的名称;

- 已收到该名称的兴趣包的传入面孔列表，以及相关信息(例如，此面孔上的兴趣到达时间);
- 兴趣数据包已转发到的输出面孔列表及相关信息(例如，在此面孔上发送兴趣的时间、在此面孔上重传兴趣的次数等);
- 条目到期的时间(同一名称收到的所有兴趣中最长的有效期)，以及
- 转发策略标签形式的任何其他特定于转发策略的信息(从ndn::fw::Tag派生的任何类)。

当前版本的ndnSIM提供了PIT抽象的模板化实现，允许可选地绑定PIT条目的数量和不同的替换策略，包括

- persistent (ndn::pit:: persistent) -如果PIT大小达到限制，新条目将被拒绝;
- random (ndn::pit:: random) -当PIT达到其极限时，随机条目(可能是新创建的条目)将从PIT中删除;
- least-recently-used (ndn::pit::Lru) -当PIT大小达到极限时，将删除最近使用最少的条目(最少输入面数的最老条目)。

所有当前的PIT实现都组织在基于尝试的数据结构中，对数据名称进行基于哈希的索引(在trie上按组件查找)和附加的时间索引(按过期时间)，以优化从PIT中删除超时的兴趣。

为每个Interest创建一个具有唯一名称的新PIT条目。当接收到具有先前已见过的名称的兴趣时，现有PIT条目的“传入面孔”列表将相应地更新，从而有效地聚合(抑制)类似的兴趣。

PIT抽象提供了以下特定于实现的操作:

- 查找:为兴趣或数据包的给定内容名称找到相应的PIT条目;
- 创建:为给定的兴趣创建一个新的PIT条目;
- 标记:移除或标记PIT入口以便移除;
- GetSize, **Begin**, **End**, **Next:**获取PIT中的表项数并遍历表项。

2.6 转发信息库(FIB)

NDN路由器的FIB与IP路由器的FIB基本相似，只是它包含名称前缀而不是IP地址前缀，并且(通常)为每个名称前缀显示多个接口。转发策略使用它来做出兴趣转发决策。

FIB (ndn:: FIB: fifiimpl)的当前实现是在基于尝试的数据结构中组织的，在数据名称上使用基于哈希的索引(在trie上按组件查找)，其中每个条目包含一个前缀和(外向)面的有序列表，通过该列表可以访问前缀。面的顺序被定义为一个综合指数，结合了面路由度量和数据平面反馈。匹配的查找以最长前缀匹配方式对可变长度前缀执行

2.6.1 FIB入口

目前，ndnSIM提供了几种方法来填充FIB表项。首先，可以使用模拟脚本为模拟设置中的每个节点手动配置fib。这种方法使用户可以完全控制哪些条目存在于哪个FIB中，并且可以很好地用于小规模模拟。然而，对于具有大型拓扑结构的模拟，它可能变得不可行的。

第二种方法是使用中央全局NDN路由控制器自动填充所有路由器的fib。当请求时(无论是在模拟运行开始之前，还是在模拟期间的任何时候)，全局路由控制器获得关于安装了NDN堆栈和所有导出前缀的所有现有节点的信息，并使用该信息计算每个节点对之间的最短路径并更新所有fib。Boost。图库(http://www.boost.org/doc/libs/release/libs/graph/)在这个计算中使用。

在当前版本中，全局路由控制器使用Dijkstra的最短路径算法(使用Face metric)，并且为每个名称前缀只安装一个出站接口。为了实验多路径兴趣转发场景，需要对全局路由控制器进行扩展，用多个条目填充每个前缀。然而，这取决于特定的模拟来定义多条目的确切基础，例如，条目是否应该表示没有公共链接的路径。我们欢迎建议和/或全局路由控制器扩展，可以在GitHub网站(https://github.com/NDN-Routing/ndnSIM)上提交。

最后，填充FIB的一个简单方法是安装默认路由(route to /)，其中包括NDN堆栈的所有可用面。例如，这种方法可以用于探索在没有任何来自路由平面的指导的情况下，Interest转发策略如何很好地找到和维护到前缀的路径的模拟。

2.7 转发策略抽象

我们的设计可以对各种类型的转发策略进行实验，而不需要修改核心组件。这一目标是通过引入转发策略抽象(ndn::ForwardingStrategy)来实现的，该策略在类事件中实现了对兴趣和数据包的核心处理

6. In addition to the longest-prefix match, next release of ndnSIM will implement several Interest selectors, including two types of ex-clude filters (ordered exclude and Bloom filters) and min/max name components filter [2].

时尚。换句话说，兴趣和数据包处理的每一步，包括内容库、PIT、FIB查找，都被表示为虚拟函数，可以在特定的转发策略实现类中重写。

更具体地说，转发策略抽象提供了以下一组可重写的动作:

- **OnInterest:**由ccnxl3协议对每个传入的Interest包调用;
- **OnData:**由ccnxl3协议为每个传入数据包调用;
- **WillErasePendingInterest:**在移除PIT条目之前触发;
- **RemoveFace:**调用删除对Face的引用(如果转发策略实现使用了任何引用);
- **DidReceiveDuplicateInterest:**在检测到接收到重复的Interest后触发;
- **DidExhaustForwardingOptions:**当转发策略耗尽所有转发选项来转发兴趣时触发;
- **FailedToCreatePitEntry:**当尝试创建PIT表项失败时触发;
- **DidCreatePitEntry:**创建PIT表项成功后触发;
- **DetectRetransmittedInterest:**在检测到重传的兴趣后触发。这甚至是可选的，只有当“DetectRetransmissions”选项在场景中启用时才会被触发。检测重传的兴趣是基于观察到与现有PIT条目匹配的新兴趣到达记录在条目传入列表中的Face。更多细节请参考源代码。
- **WillSatisfyPendingInterest:**在pending Interest对Data满意之前触发;
- **SatisfyPendingInterest:**满足pending Interest的实际过程;
- **DidSendOutData:**每次数据在Face上成功发送时触发(可能在拥塞或限速模块启用时失败);
- **DidReceiveUnsolicitedData:**每次数据到达时触发，而数据的名称没有相应的未决利息。如果启用了“CacheUnsolicited-Data”选项，那么这样的数据包将被默认的处理实现缓存。
- **ShouldSuppressIncomingInterest:**挂钩兴趣抑制逻辑;
- **TrySendOutInterest:**在实际发送Interest on a Face之前被解雇;
- **DidSendOutInterest:**在成功发送Interest on a Face后被解雇;
- **PropagateInterest:**基本的兴趣传播逻辑;
- **DoPropagateInterest**(特定于实现):特定于实现的兴趣传播逻辑。

我们预计在未来的版本中会有更多的事件被添加到转发策略抽象中。同时，可以通过类继承以面向目标的方式创建额外的事件。例如，Interest NACK扩展(详细信息请参见[5])被实现为转发策略抽象的部分专门化。

图3显示了当前可用的转发策略扩展(Nacks, greenyellow - lowred)和可以在模拟场景中使用的完整转发策略实现(Flooding, SmartFlooding和BestRoute)的部分层次结构。虽然所有当前的实现都继承自Nack扩展，实现了额外的处理和事件来检测和处理感兴趣的NACKs[5]，但NACK处理被默认故障禁用，可以使用“EnableNACKs”选项启用。

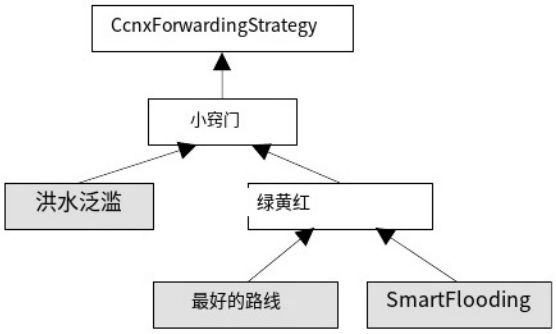

图3.。可用的转发策略(洪泛、smart -洪泛和BestRoute是完全实现的)

SmartFlooding和BestRoute的实现依赖于每个人脸状态的颜色编码，基于观测服务的数据平面反馈[5]。

- GREEN:面部工作正常(例如，如果兴趣被发送到该面部，则返回数据);
- YELLOW:Face的状态未知(例如，可能是最近添加的，或者有一段时间没有使用);
- RED:该脸不工作，不应该用于兴趣转发。

状态信息附加到FIB表项中的每个Face上，初始化为YELLOW状态。每当一个数据包作为对前一个兴趣的响应返回时，FIB表项中对应的Face被设置为GREEN颜色。每次发生错误(PIT进入超时，或者如果启用，则接收到NACK-Interest)时，Face将转回YELLOW状态。如果一张脸没有被使用足够长的时间，它就会变回YELLOW状态。当下层通知NDN堆栈出现问题(链路故障、连接错误等)时，RED状态被分配给Face。

下面的列表总结了目前可用的全转发策略实现中的处理逻辑:

- 泛洪**策略**(ndn::fw::Flooding):一个Interest报文将被转发到FIB表项中Interest前缀下的所有可用面，除了该Interest的入站面。
- 智能**泛洪策略**(ndn::fw:SmartFlooding):如果FIB表项中至少包含一个GREEN Face，则只将interest转发给最高级的GREEN Face。否则，所有YELLOW面孔将被使用

转发兴趣。RED面孔不用于兴趣转发。

该策略模式可用于不需要路由输入的仿真，数据平面可以利用兴趣包发现和维护工作路径。

- 最佳路由**策略**(ndn::fw::BestRoute)将兴趣包转发到**最高级别的**GREEN(如果可用)或YELLOW面。RED Faces不用于Interest转发。

还有一个实验性的SimpleLimits转发策略(继承了ndn::fw::BestRoute的大部分动作)，它试图通过考虑人脸选择过程中的利率限制来避免网络中的拥塞并最大化网络利用率。例如，如果排名最高的脸已经达到了它的容量——更具体地说，该脸的未决兴趣数量达到了设定的最大限制——该策略将在排名顺序中选择低于限制的下一个脸。

除了已经实现的策略之外，我们还在努力实现额外的兴趣转发策略，包括精确模拟CCNx项目实现行为的策略。

2.8 参考应用

应用程序通过ndn::AppFace实现Face抽象与系统核心交互。为了简化特定NDN应用程序的实现，ndnSIM提供了一个基本的NDN::App类，它负责创建NDN::AppFace并将其注册到NDN协议栈中，并为传入的兴趣和数据包提供默认处理。

下面列出了ndnSIM中当前可用的参考应用程序集:

- ndn::ConsumerCbr:一个应用程序，以预定义的频率生成兴趣流量(恒定速率，恒定平均速率，兴趣间隔均匀随机分布，指数随机分布等)。生成的兴趣名称包含一个可配置的前缀和一个序列号。当一个特定的兴趣在基于rtt的超时时间内(与TCP RTO相同)不被满足时，这个兴趣被重新表达。
- ndn::ConsumerBatches:一个开关样式的应用程序，在指定的模拟点生成指定数量的兴趣。名称和重传逻辑类似于ndn::ConsumerCbr应用程序。
- ndn::Producer一个简单的兴趣接收器应用程序，它用指定大小和名称的数据包回复每个传入的兴趣。

3 相关工作

在过去的几年中，人们致力于开发用于NDN架构研究的评估基础设施。

NDN项目团队现有的一项工作是在开放网络实验室(ONL)[6]上支持NDN。ONL目前包含14个可编程路由器，100多个客户端节点，通过各种功能的链路和交换机连接。每个节点和路由器都运行CCNx项目NDN实现。用户可以完全访问ONL任何节点的硬件和软件状态。还可以在detlab测试平台[7]的节点上运行和评估CCNx项目NDN实现。拥有一个可编程的非虚拟化测试平台是一个非常有价值的选择，尽管它的能力仅限于评估相对较小规模的网络。对于更大规模的实验，研究人员可能需要求助于模拟。

Rossi和Rossini[8]开发了ccnSim来评估NDN的缓存性能。ccnSim是一个可扩展的NDN块级模拟器，在omnet++框架下用c++编写，它允许在标准消费级计算机硬件上评估大规模场景(最多10-^{6}块)下的NDN性能。ccnSim的设计和实现的主要目标是运行NDN路由器内容存储的不同缓存替换策略的实验。因此，它不是现有NDN协议的完整功能实现。在当前版本的ccnSim中，PIT和FIB组件以尽可能简单的方式实现，因此无法评估不同的数据转发策略、不同的路由策略或不同的拥塞控制方案。

另一个NDN模拟器是由Muscariello和Gallo[9]在Orange Labs编写的。他们的以内容为中心的网络数据包级模拟器(CCNPL-Sim)基于SSim，这是一个实用程序库，实现了一个简单的离散事件模拟器。[10](Combined Broadcast and Content-Based routing scheme, CBCB)必须作为SSim和CCNPL-Sim之间的中间层，在一般的点对点网络上实现基于名称的路由和转发。尽管规范的NDN模型在c++中的CCNPL-Sim中完全重新实现，但该解决方案有一个缺点，即使用大多数研究人员不熟悉的自定义离散事件模拟器。此外，CBCB的强制性使用缩小了可能的实验区域，使得无法评估其他路由协议，例如OSPF- n (NDN的OSPF扩展)或双曲度量空间[11]上的路由。

Ur-bani等人采用了一种完全不同的方法。他们为NS-3模拟器内的CCNx项目NDN实现提供了直接代码执行(DCE)支持。DCE NS-3模块的总体目标是提供在NS-3模拟环境中执行现有用户空间和内核空间网络协议实现的设施。这种方法的主要优点是，模拟可以直接使用现有的未修改的CCNx代码，从而提供最大的真实感，不需要代码维护(因为新版本应该在DCE NS-3中运行，不需要太多的努力)。然而，这种方法

也引发了一些担忧。首先，包括CCNx项目代码在内的实际实现相当复杂，难以修改以探索不同的设计方法，并且包含许多与模拟评估无关的代码。其次，有一个已知的扩展问题，因为每个模拟节点必须运行一个重型DCE层和一个全尺寸的真实CCNx实现。

4 摘要

ndnSIM被设计为一组松散绑定的组件，使研究人员有机会修改或替换任何组件，而不会或很少影响ndnSIM的其他部分。我们的模拟器提供了一组参考应用程序和助手类，允许在许多不同的场景下评估NDN协议的各个方面。ndnSIM的第一个版本于2012年6月公开发布，有关该版本的更多详细信息和附加文档可在ndnSIM网站http://irl.cs.ucla.edu/ndnSIM/上获得。

我们希望社区发现ndnSIM是有用的，我们期待社区的反馈来帮助进一步改进它。

参考文献

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in *Proceedings of ACM CoNEXT*, 2009.

[2] (2012, May) Project CCNx. [Online]. Available: http://www.ccnx.org

[3] (2012, February) The NDN project testbed. [Online]. Available: http://www.named-data.net/testbed.html

[4] (2012, May) ns-3. [Online]. Available: http://www.nsnam.org/

[5] Y. Cheng, A. Afanasyev, I. Moiseenko, B. Zhang, L. Wang, and L. Zhang, “Smart forwarding: A case for stateful data plane,” Tech. Rep. NDN-0002, May 2012.

[6] L. Zhang et al., “Named data networking (NDN) project 2010 - 2011 progress summary,” PARC, http://www. named-data. net/ndn-ar2011. html, Tech. Rep., November 2011.

[7] “DETER network security testbed,” https://www.isi.deterlab.net.

[8] D. Rossi, G. Rossini, “Caching performance of content centric networksunder multi-path routing (and more),” Telecom ParisTech, Tech. Rep., 2011.

[9] L. Muscariello. (2011) Content centric networking packet level simulator. Orange Labs. [Online]. Available: http://perso. rd. francetelecom. fr/muscariello/sim.html

[10] A. Carzaniga, M.J. Rutherford, and A.L. Wolf, Ed., *A Routing Scheme for Content-Based Networking*. IEEE INFOCOM, March 2004.

[11] CAIDA. Caida’s role in the ndn. [Online]. Available: http: //www.caida.org/projects/ndn-fia/

[12] F. Urbani, W. Dabbous, and A. Legout. (2011, Novem-ber) NS3 DCE CCNx quick start. INRIA. [Online]. Available: http://www-sop.inria.fr/members/Frederic.Urbani/ns3dceccnx/index.html