**计算机通信网络协议与安全**

**实验报告**

1. **实验信息**

实验平台：Ubuntu20.04

模拟环境：NS3

1. **实验需求：**

要求：

基于NS3仿真平台的CSMA NetDevice模块，验证CSMA的性能，画出网络吞吐量S随网络数据到达率G之间的关系图。

提交：

（1）仿真代码；

（2）仿真结果讨论（包括仿真参数配置，S-G关系曲线）

1. **提交内容：**

2.1 仿真结果讨论

本次实验设计：

// Network topology

//

//       n0 -> n1   n2 -> n3

//       |     |     |    |

//       ==================

//              LAN

共涉及4个节点，其中n0-n4均处于同于使用CSMA的网段，并且n0向n1传输数据，同时n2向n3传输数据，由此导致传输冲突，借此来测试CSMA协议的有效性。

其中本次实验设置，n0和n2同时在1s时开始进行报文发送，n1和n3同时开启报文接受服务。n0和n2的报文发送服务会在10s时结束，而n1和n3会继续完成报文的传输，保证结果的有效性。

本次仿真参数配置、检测结果如下表：

实验参数设置于结果记录表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据 实验 | 参数 | 传输速率 | CSMA延迟 | 上传速率 | 报文长度 | 平均吞吐量 | 到达率 | 平均吞吐量 |
| 单位 | Mb/s | ms | kb/s | b | b/s | p/s | p/s |
| 1 |  | 5 | 2 | 32 | 128 | 39.12 | 0.25 | 0.31 |
| 2 |  | 5 | 2 | 64 | 128 | 78.50 | 0.50 | 0.61 |
| 3 |  | 5 | 2 | 96 | 128 | 117.82 | 0.75 | 0.92 |
| 4 |  | 5 | 2 | 128 | 128 | 157.11 | 1.00 | 1.23 |
| 5 |  | 5 | 2 | 160 | 128 | 196.49 | 1.25 | 1.54 |
| 6 |  | 5 | 2 | 192 | 128 | 235.57 | 1.50 | 1.84 |
| 7 |  | 5 | 2 | 224 | 128 | 257.60 | 1.75 | 2.01 |
| 8 |  | 5 | 2 | 256 | 128 | 256.25 | 2.00 | 2.00 |
| 9 |  | 5 | 2 | 288 | 128 | 245.66 | 2.25 | 1.92 |

通过上表的实验结果，可以得到S-G（吞吐量和到达率）关系图，如下图所示：

由此可见，随着到达率的上升，吞吐量在到达率低时逐渐上升，在到达率高时逐渐趋于平缓，不再变化，甚至有下降的趋势；

2.2 仿真代码

#include <iostream>

#include <fstream>

#include "ns3/core-module.h"

#include "ns3/network-module.h"

#include "ns3/csma-module.h"

#include "ns3/applications-module.h"

#include "ns3/internet-module.h"

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE("CsmaOneSubnetExample");

int  main(int argc, char \*argv[])

{

    CommandLine cmd;

    cmd.Parse(argc, argv);

    NodeContainer nodes;

    nodes.Create(4);

    const char rate[9] = "128kb/s";

    CsmaHelper csma;

    csma.SetChannelAttribute("DataRate", DataRateValue(5000000));

    csma.SetChannelAttribute("Delay", TimeValue(MilliSeconds(2)));

    NetDeviceContainer devices = csma.Install(nodes);

    InternetStackHelper internet;

    internet.Install(nodes);

    NS\_LOG\_INFO("Assign IP Addresses.");

    Ipv4AddressHelper ipv4;

    ipv4.SetBase("114.2.19.0", "255.255.255.0");

    Ipv4InterfaceContainer interfaces = ipv4.Assign(devices);

    OnOffHelper onoffApp("ns3::UdpSocketFactory", Address(InetSocketAddress(interfaces.GetAddress(1), 666)));

    onoffApp.SetConstantRate(DataRate(rate));

    onoffApp.SetAttribute("PacketSize", UintegerValue(128));

    ApplicationContainer app = onoffApp.Install(nodes.Get(0));

    app.Start(Seconds(1.0));

    app.Stop(Seconds(10.0));

    PacketSinkHelper sink("ns3::UdpSocketFactory", Address(InetSocketAddress(Ipv4Address::GetAny(), 666)));

    app = sink.Install(nodes.Get(1));

    app.Start(Seconds(0.0));

    onoffApp.SetAttribute("Remote", AddressValue(InetSocketAddress(interfaces.GetAddress(3), 666)));

    app = onoffApp.Install(nodes.Get(2));

    app.Start(Seconds(1.0));

    app.Stop(Seconds(10.0));

    app = sink.Install(nodes.Get(3));

    app.Start(Seconds(0.0));

    AsciiTraceHelper ascii;

    csma.EnableAsciiAll(ascii.CreateFileStream("csma.tr"));

    csma.EnablePcapAll("csma", true);

    Simulator::Run();

    Simulator::Destroy();

}