CS144：计算机网络导论

分组交换

CS144, Stanford University

大纲

1. 端到端延迟
2. 排队延迟
3. 简单确定性队列模型
4. 示例

传播延迟

传播延迟 `t_l`: 一个比特以传播速度 c 在链路上传输所需的时间。

t_l = l / c

示例: 一个比特在传播速度为 2 x 10^8 m/s 的光纤中传输1,000km需要5ms。

通过链路发送一个分组的总时间

从第一个比特发送到最后一个比特到达的时间。

$t_p = t_t + t_l = p/r + l/c$

示例: 一个100bit的分组以 10Mb/s 的速率通过 1km 链路发送需要 10 + 5 = 15ms。

端到端延迟

从 A 到 B 的路径, 假设交换机对分组进行存储转发。

Q: 长度为 p 的分组从 A 到 B 需要多长时间? (从第1个比特发送到最后一个比特到达)

端到端延迟 t = Sum(p/r_i + l_i/c)

存储转发时序图

分组通过多个交换机 (S1, S2, S3) 的时序图。

端到端延迟 = Sum(p/r_i + l_i/c)

带排队延迟的端到端延迟

其他分组可能在队列中等待, 增加了额外延迟。

端到端延迟 = Sum(p/r_i + l_i/c + Q_i(t))

分组延迟变化

Stanford-Princeton: 4,000 km, 变化约50ms

Stanford-Tsinghua: 10,000 km, 变化约200ms

(累积分布函数图 CDF)

路由器队列的简单模型

路由器队列的简单模型

A(t): 截止时间 t 的累计到达数

D(t): 截止时间 t 的累计离开数

Q(t): 时间 t 时队列中的分组数

来自不同输入端口的到达分组

性质:

1. A(t), D(t) 单调不减
2. A(t) >= D(t)

队列的简单模型

A(t): 截止时间 t 的累计到达字节数。

D(t): 截止时间 t 的累计离开字节数。

Q(t): 队列占用量。

性质:

1. A(t), D(t) 单调不减
2. A(t) >= D(t)

队列的简单模型

队列占用量: Q(t) = A(t) - D(t)

排队延迟 d(t): 在时间 t 到达的字节在队列中花费的时间, 假设队列按先到先服务 (FCFS) 的方式服务。

示例（存储转发）

每秒, 一个 500 bit 的分组以 10,000b/s 的速率到达队列。最大离开速率为 1,000b/s。队列的平均占用量是多少?

解: 在每个重复的1s周期内, 队列以 10,000b/s 的速率填充0.05s, 然后以 1,000b/s 的速率排空0.5s。在前0.55s内, 平均队列占用量为 250 bits。队列在每个周期中空闲0.45s, 因此平均队列占用量为 (0.55 * 250) + (0.45 * 0) = 137.5 bits。

示例（“快速转发”）

每秒，一个 500 bit 的分组以 10,000b/s 的速率到达队列。最大离开速率为 1,000b/s。队列的时间平均占用量是多少?

解：在每个重复的1s周期内，队列以 10,000b/s 的速率在前0.05s内填充到 500-50=450 bits，然后以 1,000b/s 的速率排空0.45s。在前0.5s内，平均队列占用量为 225 bits。队列在每个周期中空闲0.5s，因此平均队列占用量 = 0.5 x 225 + 0.5 x 0 = 112.5 bits。

如果某些分组比其他分组更重要呢?

默认情况下，交换机和路由器使用 FIFO（即 FCFS）队列

- 缓冲区大小: B
- 分组从不同的入口端口到达
- 服务速率: R
- 分组离开到不同的目的地

默认情况下，交换机和路由器使用 FIFO（即 FCFS）队列

(图示变体)

某些分组更重要

例如:

1. 维持网络运行的控制流量 (例如携带路由表更新的分组)
2. 来自特定用户的流量 (例如付费更多的客户)
3. 属于特定应用的流量 (例如视频会议)
4. 发往/来自特定 IP 地址的流量 (例如紧急服务)
5. 时间敏感的流量 (例如时钟更新)

流 (Flows)

在讨论优先级时, 讨论共享一组公共属性的分组“流”是很方便的。例如:

1. 属于同一个 TCP 连接的分组流 — 由元组标识: TCP 端口号、IP 地址、TCP 协议
2. 发往 Stanford 的分组流 — 由属于前缀 171.64/16 的目的 IP 地址标识
3. 来自 Google 的分组流 — 由属于 Google 拥有的前缀集合的源 IP 地址标识
4. 使用 HTTP 协议的 Web 分组流 — 由 TCP 端口号 = 80 的分组标识
5. 属于金级服务客户的分组流 — 通常通过标记 IP TOS (服务类型) 字段来标识

大纲

1. 严格优先级 (Strict Priorities)
2. 加权优先级与速率保证

严格优先级

- 高优先级流
- 低优先级流

严格优先级

高优先级流

低优先级流

“严格优先级” 意味着只有在所有更高优先级的队列都为空时, 才会服务某个队列。

严格优先级：需要注意的事项

1. 严格优先级可以与任意数量的队列一起使用。
2. 严格优先级意味着只有在所有更高优先级的队列都为空时, 才会服务某个队列。
3. 最高优先级流“看到”的是一个没有低优先级流量的网络。
4. 高优先级流可能永久阻塞低优先级流。尽量限制高优先级流量的量。
5. 如果无法控制高优先级流量的量, 可能无法很好地工作。
6. 或者如果你真的想要加权 (而不是严格) 优先级。

如何给予加权（而不是严格）优先级?

试图平等对待流

试图平等对待流

虽然每个流以相同的分组速率发送, 但数据速率远不相等。

逐比特调度流

逐比特调度流

现在每个流以相同的数据速率发送, 但我们不再有“分组交换”了。

我们能否结合两者的优点?

即: 分组交换, 但带有逐比特计算?

公平队列（Fair Queueing）

分组按照它们在逐比特方案中完成的顺序发送。

问: 这是否给予了公平 (即相等) 的数据速率份额?

是的!

1. 可以证明, 使用公平队列时分组的离开时间不会比逐比特调度晚超过 L_max/R 秒。其中 L_max 是最大长度分组, R 是出口链路的数据速率。

2. 在极限情况下, 两个流获得相等的数据速率份额。

3. 该结果可扩展到共享链路的任意数量的流。

[1] "Analysis and Simulation of a Fair Queueing Algorithm" Demers, Keshav, Shenker. 1990.

如果我们想给每个流不同的链路份额呢?

即: 加权公平份额。

加权公平队列（Weighted Fair Queueing）

与之前一样，分组按照它们在逐比特方案中完成的顺序发送。

加权公平队列（WFQ）

对于任意数量的流和任意分组大小的组合:

1. 使用加权逐比特方案确定每个分组的离开时间。
2. 按离开时间递增的顺序转发分组。

流 i 保证获得至少 phi_i * R 的速率

加权公平队列（WFQ）

将到达不同入口端口的分组分类为流，然后通过分组调度器进行调度。

流 i 保证获得至少 phi_i * R 的速率

总结

1. FIFO 队列是自由竞争的: 没有优先级, 没有保证速率。

2. 严格优先级: 高优先级流量“看到”的是一个没有低优先级流量的网络。如果高优先级流量有限, 这很有用。

3. 加权公平队列 (WFQ) 允许我们通过按逐比特完成时间的顺序调度流, 从而给每个流一个保证的服务速率。

我们能否保证分组通过分组交换机网络的延迟?

CS144, Stanford University

延迟保证：直觉

以下值是固定的 (或在我们的控制下): p, c, l_i 和 r_i。

如果我们知道 Q_1(t)、Q_2(t) 和 Q_3(t) 的上界, 那么我们就知道端到端延迟的上界。

队列延迟上界

Q(t) 的上界为 b, 服务速率为 r, 则 Q(t) <= b/r

示例: 如果一个分组到达大小为 100万比特的 FIFO 队列, 队列服务速率为 1Gb/s, 则该分组保证在 10^6 / 10^9 = 1ms 内离开。

延迟保证：直觉

端到端延迟上界公式，将每个队列的最大延迟累加。

为什么这只是直觉……

1. 没有告诉我们当 r_2 < r_1 时会发生什么。分组会被丢弃吗?
2. 将共享队列的所有分组视为一个大流; 它没有为每个流给出不同的端到端延迟。

Q: 如何为每个单独的流给出延迟上界?

加权公平队列（WFQ）

将到达不同入口端口的分组分类为流。

流 i 保证获得至少 phi_i * R 的速率

加权公平队列（WFQ）

每个流有缓冲区大小 b_i。

流 i 的延迟 <= b_i / (phi_i * R)

流 i 保证获得至少 phi_i * R 的速率

约束端到端延迟

约束端到端延迟

1. 为该流分配大小为 b 的队列
2. 分配 WFQ 服务速率 phi * R

单个分组的端到端延迟上界公式。

如果流的两个分组背对背进入网络呢？（“突发”）

1. 如果分组相距很远, 队列在第二个到达之前就排空了第一个。一切顺利, 延迟公式成立。
2. 如果分组在“突发”中紧密相连, 它们可能以超过 phi*R 的速度到达, 队列可能溢出, 导致分组丢弃。
3. 在某些情况下这可能没问题。但如果我们想约束所有分组的端到端延迟, 就需要处理突发。怎么办?

漏桶调节器 (Leaky Bucket Regulator)

限制“突发性”

以恒定速率 r 生成令牌

令牌桶大小: s

分组输入 -> 分组输出

当且仅当桶中有令牌时才发送分组

漏桶调节器 (Leaky Bucket Regulator)

在任意长度为 t 的时间段内可发送的比特数受以下约束: sigma + rho * t

它也被称为 "(sigma, rho) 调节器"

漏桶调节器

很酷的定理: 如果到达队列的流量是 (sigma, rho)-约束的, 并且队列以 phi*R > rho 的速率服务且 b > sigma, 那么离开的流量也是 (sigma, rho)-约束的。这意味着到达下一个路由器的流量也是 (sigma, rho)-约束的。

如果 phi*R > rho 且 b > sigma, 则该流中所有分组通过第一个路由器的延迟 <= b / (phi*R)

综合在一起

如果 phi*R > rho 且 b > sigma, 则每个分组的端到端延迟上界可以计算。

换句话说

如果我们设置 b > sigma, 且 phi_i * R > rho, 则可以计算端到端延迟上界。

最终延迟上界 = Sum(p/r_i + l_i/c) + 3*sigma/rho

一个例子

Q: 在下面的网络中, 我们希望给一个应用流提供 10Mb/s 的速率和小于 4.7ms 的端到端延迟(1,000字节分组)。漏桶调节器应该使用什么样的 sigma 和 rho 值? 路由器需要什么样的服务速率和缓冲区大小? (假设传播速度 c = 2 x 10^8 m/s)

A: 固定延迟分量为 (120km/c) + 8,000bits * (1/10^9 + 1/(100*10^6) + 1/10^9) = 0.7ms, 留给路由器队列的延迟为 4ms。将 2ms 延迟分配给每个路由器, 即每个路由器的队列不超过 2ms x 10Mb/s = 20,000bits (2500字节)。因此, 主机 A 中的漏桶调节器应该有 rho = 10Mb/s 和 sigma <= 20,000bits。每个路由器的 WFQ 应设置为 phi_i * R >= 10Mb/s, 队列容量至少 2500字节。

实践中

虽然几乎所有网络设备都实现了 WFQ (甚至你家的 WiFi 路由器可能也实现了!), 但公共网络并不提供控制端到端延迟的服务。

为什么? - 它需要所有路由器从端到端的协调。 - 在大多数网络中, 过度配置和优先级的组合已经足够好了。

总结

1. 如果我们知道队列的大小和服务速率, 就可以约束通过它的延迟。
2. WFQ 允许我们选择队列的服务速率。
3. 有了以上两个观察, 如果没有分组被丢弃, 我们可以控制端到端延迟。
4. 为了防止丢弃, 我们可以使用漏桶调节器来控制进入网络的流的“突发性”。