Queueing Strategien für Router

Queueing Strategien (1/2)

- Pro Router: Jeder Link (oder jede Kante) e besitzt eine Schlange s_e, in der Pakete auf ihre Beförderung über Kante e warten.
- Um die Notation zu vereinfachen, nimm an, dass in jedem Schritt nur ein Paket über eine Kante befördert werden kann.
- Die Queueing-Strategie muss das zu transportierende Paket aus den wartenden Paketen auswählen.

Populäre Queueing Strategien:

- **FIFO** (First-In-First-Out).
- In **Fair-Queueing** wird für jede Kante e **und** jeden Absender a eine Schlange $s_{e,a}$ eingerichtet.
  - Die Schlangen werden in Round-Robin Manier durchlaufen.
  - Die ausgewählte Schlange wird nach dem FIFO-Prinzip bearbeitet.
  - Als Konsequenz: Absender mit geringem Paketverkehr werden nicht durch Absender mit großem Paketverkehr benachteiligt.

Queueing Strategien (2/2)

Die zentrale Frage: Hat die Wahl der Queueing Strategie Einfluß auf die Effizienz des Paket-Transports?

- **Farthest-To-Go** (FTG): Das Paket, das noch die meisten Kanten zu durchlaufen hat, wird gewählt.
- **Nearest-To-Source** (NTS): Das Paket, das bisher die wenigsten Kanten durchlaufen hat, wird gewählt.
- **Longest-In-System** (LIS): Das Paket, das die meiste Zeit im System verbracht hat, wird gewählt.
- **Shortest-In-System** (SIS): Das Paket, das die kürzeste Zeit im System verbracht hat, wird gewählt.
- **Nearest-To-Go** (NTG): Das Paket, das die wenigsten Kanten zu durchlaufen hat, wird gewählt.
- **Farthest-From-Source** (FFS): Das Paket, das bisher die meisten Kanten durchlaufen hat, wird gewählt.

Das Queueing Modell

Ein Worst-Case Modell

Was bedeutet „effizienter Paket-Transports“?

- Ein **statistischer Ansatz** aus der Theorie der Warteschlangen legt eine Verteilung zugrunde mit der Pakete emitiert werden.
  - Das erwartete Verhalten und die Varianz werden gemessen.
  - Aber welche Verteilung modelliert das tatsächliche Paketaufkommen am besten?
- Wir wählen stattdessen einen **Worst-Case Ansatz**:
  - Ein bösartiger Gegner versucht, die Anzahl der sich im System befindlichen Pakete „hochzuschaukeln“.
  - Wir fragen, ob eine vorgegebene Queuing Strategie jeden Gegner in jedem Graphen in Schach halten kann.

Sollen wir jeden Gegner erlauben?

(r, b)-Gegner

G sei ein gerichteter Graph.

- Ein $(\mathbf{r}, \mathbf{b})$-**Gegner** $\mathcal{G}$ darf für jeden Knoten Pakete injizieren und deren Wege festlegen.
  - Für **jedes Zeitintervall I** und für **jede Kante e** dürfen höchstens
    $$\mathbf{r} \cdot |\mathbf{I}| + \mathbf{b}$$
    der während I injizierten Pakete die Kante e in ihrer Wegbeschreibung besitzen.
  - r heißt die **Injektionsrate** und b die maximale **Burstiness**.
    - Der Gegener darf Wege vorschreiben, aber keine Kante überlasten.
    - Der Gegner kann über eine Kante e laufende Pakete in gleichmäßiger Geschwindigkeit mit bis zu b Ausnahmen injizieren.
- Eine Kante transportiert nur ein Paket:
  - Ein (r, b)-Gegner testet die Queueing-Strategie für $\mathbf{r} < \mathbf{1}$ nahe an der maximal tolerierbaren Auslastung,
  - Eine Injektionsrate von $\mathbf{r} > \mathbf{1}$ ist unfair, da ein Hochschaukeln nicht verhindert werden kann.

Stabilität

- Eine Queueing-Strategie ist **gegen einen Gegner** $\mathcal{G}$ auf dem Graph G **stabil**, wenn
  - beginnend mit einem leeren System,
  - zu jedem Zeitpunkt die Anzahl der im System verbleibenden Pakete durch eine nur von G abhängende Konstante C_G beschränkt ist.
- Eine Queueing-Strategie ist **stabil**, wenn die Strategie **gegen jeden $(\mathbf{r}, \mathbf{b})$-Gegner** mit Injektionsrate $r < 1$ auf **jedem gerichteten Graphen** stabil ist.

Welche Queueing Strategien sind stabil, welche instabil?

Wie produziert man einen Datenstau?

- Wähle N knoten-disjunkte, gerichtete Wege $W_{N-1}, \ldots W_0$ der Längen $2 \cdot (N-1), 2 \cdot (N-2), \ldots, 2, 0$.
  - Weg W_i endet im Knoten u_i, der den Knoten u als einzigen Nachfolger besitzt.
- Knoten **u** seinserseits besitzt nur den Nachfolger **v**.

Ein Datenstau für Kante (u, v) bei Injektionsrate $r = \frac{1}{2}$:

1. Zuerst injiziere ein Paket in den ersten Knoten von W_{N-1},
2. dann im dritten Schritt ein Paket in den ersten Knoten von W_{N-2} etc.
3. Und wenn ein Paket in den einzigen Knoten von W_0 injiziert wird?

Eine **Greedy Queueing-Strategie**, die Pakete bei nicht-leerer Schlange weiterleitet, hat N Pakete gleichzeitig in den Knoten u gepumpt: Alle N Pakete warten auf ihren Transport über Kante $(\mathbf{u}, \mathbf{v})$.

Stabilität und Zustellzeit

- Wenn eine Queueing-Strategie gegen einen Gegner stabil ist, dann besitzen alle Schlangen zu jedem Zeitpunkt eine Höchstzahl von Paketen.
- Bedeutet Stabilität aber auch, dass Pakete in beschränkter Zeit zugestellt werden?

- Für $r < 1$ wird jede Schlange in einem genügend langen Zeitfenster entleert, denn
  - wir füllen mit Geschwindigkeit $r < 1$,
  - entleeren aber mit Geschwindigkeit 1.

Stabilität beschränkt die maximale Schlangengröße und deshalb sollte auch die Zustellzeit beschränkt sein.

Die fundamentale Beobachtung

Für eine Greedy-Strategie und die Kante e irgendeines Graphen:

Wenn es zum Zeitpunkt t genau $\mathbf{k-1}$ Pakete im System gibt, die die Kante e in ihrer Wegbeschreibung besitzen, dann wird die Schlange von e während der nächsten $\mathbf{\frac{k+b}{1-r}}$ Schritte entleert.

- Betrachte das Zeitfenster $Z = [t+1, t+\frac{k+b}{1-r}]$.
- Erreicht Paket $\mathcal{P}$ die Schlange von e im Zeitfenster Z,
  - dann war $\mathcal{P}$ entweder zum Zeitpunkt t bereits im System oder
  - ist eines von höchstens $r \cdot \frac{k+b}{1-r} + b$ später injizierten Paketen.
- Insgesamt möchten höchstens

$$k - 1 + r \cdot \frac{k+b}{1-r} + b = \frac{(1-r)\cdot(k+b-1)}{1-r} + r \cdot \frac{k+b}{1-r} < \frac{k+b}{1-r}$$

Pakete die Kante e im Zeitfenster Z durchlaufen:
Die Behauptung folgt, da wir nur Greedy Strategien betrachten.

Schlangengröße und Zustellzeit

Für $r < 1$ und einen beliebigen gerichteten Graphen $G = (V, E)$:

(a) Wenn $k \cdot |E| > b$ und wenn eine Schlange nie mehr als k Pakete besitzt, dann wird jedes Paket, das mit einem Weg der Länge $\leqslant d$ injiziert wurde, nach höchstens $\frac{2 \cdot k \cdot d \cdot |E|}{1-r}$ Schritten zugestellt.

(b) Wenn jedes Paket nach höchstens s Schritten zugestellt wird, dann befinden sich stets höchstens s Pakete in einer Schlange.

(a) Höchstens $k \cdot |E|$ Pakete befinden sich im System.
- Eine beliebige Schlange S wird irgendwann während der nächsten $\frac{k \cdot |E| + 1 + b}{1-r} \leqslant \frac{2 \cdot k \cdot |E|}{1-r}$ Schritte geleert.
- Jedes Paket überquert mindestens eine Kante innerhalb von $\frac{2 \cdot k \cdot |E|}{1-r}$ Schritten und wird nach $\leqslant \frac{2 \cdot k \cdot d \cdot |E|}{1-r}$ Schritten zugestellt.

(b) Wenn eine Schlange S mehr als s Pakete besitzt, dann wird das niedrigst-rangige Paket in S für mehr als s Schritte aufgehalten.

Shortest-In-System

Shortest-In-System (1/3)

Setze $\mathbf{k_1 = b}$ und $\mathbf{k_{j+1} = \frac{k_j+b}{1-r}}$.

Sei e eine beliebige Kante und $\mathcal{P}$ ein Paket, das e durchlaufen möchte.

Wenn $\mathcal{P}$ die Schlange seiner $j$ten Kante erreicht, dann gibt es höchstens $\mathbf{k_j - 1}$ andere Pakete mit mindestens so hoher Priorität im System, die auch e in ihrer Wegbeschreibung besitzen.

Induktion nach j:

- $\mathcal{P}$ habe seine erste Schlange erreicht:

  Nur die höchstens $b-1$ vielen gleichzeitig mit $\mathcal{P}$ injizierten Pakete haben eine gleichhohe Priorität.

Shortest-In-System (2/3)

$\mathcal{P}$ habe gerade die Schlange der $j$ten Kante e_j erreicht.

- Nach Induktionsannahme: Höchstens $k_j - 1$ Pakete besitzen $e = e_j$ auch in ihrer Wegbeschreibung und haben eine mindestens so hohe Priorität wie $\mathcal{P}$.
- Also überquert $\mathcal{P}$ die $j$te Kante nach höchstens $\frac{k_j+b}{1-r}$ Schritten und erreicht damit die Schlange der Kante e_{j+1}.
  - In der Wartezeit können höchstens $r \cdot \frac{k_j+b}{1-r} + b$ Pakete injiziert werden, die e durchlaufen möchten.
  - Wenn $\mathcal{P}$ die Schlange von e_{j+1} erreicht, haben somit höchstens

$$\leqslant k_j - 1 + r \cdot \frac{k_j + b}{1 - r} + b = (1 - r + r) \cdot \frac{k_j + b}{1 - r} - 1 = k_{j+1} - 1,$$

um e rivalisierende Pakete eine mindestens gleichhohe Priorität.

- Wenn d die Länge eines längsten Weges ist, dann besitzt keine Schlange mehr als k_d Pakete
- und kein Paket verweilt für mehr als $(d \cdot b + \sum_{i=1}^{d} k_i)/(1-r)$ Schritte im System.

- Es ist $k_j \leqslant k_d$ für alle j: Zu keinem Zeitpunkt warten mehr als k_d Pakete in der Schlange zur Kante e.
- Ein Paket überquert seine $j$te Kante nach $\leqslant (k_j + b)/(1-r)$ Schritten: Auch die obere Schranke für die Verweildauer folgt.

Wenn alle Wege logarithmischer Länge haben, dann
verweilt jedes Paket nur für polynomiell viele Schritte im System.

Farthest-To-Go

- Sei m die Kantenzahl von G und sei d die Länge eines längsten einfachen Weges von G.
- Setze $k_i = 0$ für $i > d$ und $k_i = m \cdot k_{i+1} + m \cdot b$ für $1 \leqslant i \leqslant d$.

- Zeige durch Rückwärts-Induktion: Die Anzahl der Pakete, die noch **mindestens** i Kanten traversieren müssen, ist durch k_i beschränkt.
- Die Behauptung ist für $i > d$ trivial, denn dann ist $k_i = 0$.

Die Stabilität von Farthest-To-Go ist eine einfache Konsequenz.

Farthest-To-Go (2/3)

Sei e eine beliebige Kante.

- $X_i(e, t)$ ist die Menge der Pakete in der Schlange von e zum Zeitpunkt t, die noch mindestens i Kanten traversieren müssen.
- Sei $t^* < t$ der letzte Zeitpunkt, zu dem $X_i(e, t^*)$ leer war. (Da das System anfänglich leer ist, existiert t^*.)

Wie gelangt ein Paket $\mathcal{P}$ in die Menge $X_i(e, t)$?

- Entweder wurde $\mathcal{P}$ nach Zeitpunkt t^* injiziert
  - $\mathcal{P}$ ist eines von höchstens $r \cdot (t - t^*) + b$ Paketen: $\leqslant r \cdot (t - t^*) + b$ im Zeitraum $[t^* + 1, t]$ injizierte Pakete dürfen e durchlaufen.
- oder $\mathcal{P}$ hatte zum Zeitpunkt t^* die Schlange von e noch nicht erreicht und muss deshalb mindestens $i + 1$ Kanten traversieren.
  - $\mathcal{P}$ ist nach Induktionsannahme eines von höchstens k_{i+1} Paketen.

Im Zeitfenster $[t^* + 1, t]$ muss **stets** ein Paket mit noch mindestens i zu traversierenden Kanten die Schlange von e verlassen. Warum?

- Wir verwenden Farthest-To-Go
- und t^* ist der letzte Zeitpunkt, zu dem $X_i(e, t^*)$ leer war.

- Als Konsequenz:

$$
\begin{aligned}
|X_i(e,t)| &\leqslant k_{i+1} + r \cdot (t - t^*) + b - (t - t^*) \\
&= k_{i+1} - (1 - r) \cdot (t - t^*) + b \\
&\leqslant k_{i+1} + b.
\end{aligned}
$$

- Es gibt $\leqslant m \cdot k_{i+1} + m \cdot b = k_i$ Pakete, die noch mindestens i Kanten traversieren müssen: Der Induktionsschritt ist gezeigt.

Instabile Strategien

Instabilität und die Anfangskonfiguration

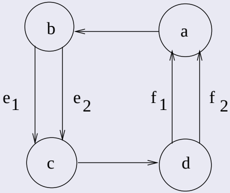
- Die Queueing-Strategie Q sei instabil gegen einen (r, b)-Gegner $\mathcal{G}$ auf dem Graphen G für irgendeine Anfangskonfiguration.
- Dann gibt es Graphen G^* und (r, b)-Gegner $\mathcal{G}^*$, so dass Q instabil auf G^* gegen $\mathcal{G}^*$ für die leere Anfangskonfiguration ist.

- Für jeden Knoten v: Klebe einen Baum T_v an Knoten v.
  - Wenn Knoten v ursprünglich p_v Pakete erhalten hat, dann erhält die Wurzel genau p_v Kinder, die „Anführer“ einer langen Kette sind.
  - Der Baum T_v muss so gebaut sein, dass
    1.) in jedem $\frac{1}{r}$ten der ersten $\frac{1}{r} \cdot \sum_{v \in V} p_v$ Schritte genau ein Paket in ein Blatt injiziert wird,
    2.) sich Pakete erst an der Wurzel ihres Baums treffen und
    3.) die Pakete eines jeden Baums unter jeder Greedy-Strategie die Wurzel zum selben Zeitpunkt erreichen.
- Wenn alle Pakete ihre Wurzel zum Zeitpunkt t^* erreichen, dann übernimmt der alte Gegner $\mathcal{G}$.

FIFO

FIFO: Die Anfangskonfiguration

Für $r \geqslant 0.85$ gibt es eine Anfangskonfiguration auf der FIFO gegen einen $(r, 0)$-Gegner auf dem Graphen



instabil ist.

- Am Anfang wartet eine Menge W von n Paketen in der Schlange von (a, b).

  Die Pakete besitzen b als Zielknoten.

- Am Ende warten $\geqslant n + 1$ Pakete in der Schlange von (c, d).

1.) Während der ersten n Schritte wird eine Menge X von $r \cdot n$ Paketen mit Wegbeschreibung $(a, b), e_2, (c, d)$ im Knoten a eingefügt.
   - Diese Pakete müssen in der Schlange von (a, b) warten, da die n Pakete aus W nach der FIFO-Regel Vorfahrt haben.

2.) In den nächsten $r \cdot n$ Schritten wird eine Menge Y von $r^2 \cdot n$ Paketen mit Wegbeschreibung $(a, b), e_1, (c, d)$ in a eingefügt.
   - Diese Pakete müssen den Transport aller Pakete aus X über Kante (a, b) abwarten.

   Zusätzlich wird eine Menge Z von $r^2 \cdot n$ Paketen mit Wegbeschreibung e_2 im Knoten b eingefügt:
   - Diese Pakete halten die Pakete aus X auf.
   - Die Pakete aus X haben eine Dichte von $\frac{1}{r+1}$ in der e_2-Schlange:
     - ★ In K Schritten werden $r \cdot K$ viele Z-Pakete neu injiziert,
     - ★ K Pakete aus X sind in der Schlange angekommen und
     - ★ insgesamt sind $(r + 1) \cdot K$ viele X- und Z-Pakete angekommen.

Der $(r, 0)$-Gegner (2/2)

Die Situation nach den ersten $n + r \cdot n$ Schritten:

- Nur $r \cdot n \cdot \frac{1}{r+1}$ X-Pakete kreuzen die Kante e_2.
- Die restlichen $r \cdot n - \frac{r \cdot n}{r+1} = \frac{r^2 \cdot n}{r+1}$ Pakete aus X verbleiben in b.
- Alle Y-Pakete warten im Knoten a, fließen aber in den nächsten Schritten über die Kanten (a, b) und e_1 in den Knoten c.

3.) Während der nächsten $r^2 \cdot n$ Schritte treffen sich die $\frac{r^2 \cdot n}{r+1}$ X-Pakete und die $r^2 \cdot n$ Y-Pakete im Knoten c.

- Füge $r^3 \cdot n$ Pakete mit Wegbeschreibung (c, d) in Knoten b ein.
- Insgesamt rivalisieren $r^2 \cdot n + \frac{r^2 \cdot n}{r+1}$ Pakete aus $X \cup Y$ mit den gerade injizierten $r^3 \cdot n$ Paketen um die Kante (c, d).
- Da nur $r^2 \cdot n$ Pakete abfließen können, warten $\geqslant r^3 \cdot n + \frac{r^2 \cdot n}{r+1}$ Pakete in der (c, d)-Schlange von (c, d).

Es ist $r^3 \cdot n + \frac{r^2 \cdot n}{r+1} > n$, falls $r \geqslant 0.85$.

FIFO: Zusammenfassung

FIFO erlaubt, dass die Störpakete aus Z viele X-Pakete aufhalten.

- FIFO ist selbst für beliebig kleine Injektionsraten $r > 0$ instabil.
- Ein „schleichender“ Angriff kann erfolgreich durchgeführt werden, solange der Angreifer die Wegbeschreibung der Pakete kontrolliert:

  Selbst nur langsam eingefügte Pakete können nicht schnell genug vom System befördert werden.

Zusammenfassung und offene Fragen

- Farthest-To-Go, Nearest-To-Source, Longest-In-System und Shortest-In-System sind stabil.

  Farthest-To-Go und Nearest-To-Source sind sogar für $r = 1$ stabil.
- FIFO, Nearest-To-Go, Farthest-From-Source sowie Last-In-First-Out sind hingegen instabil.

- Besitzen Farthest-To-Go, Nearest-To-Source und Shortest-In-System **polynomielle** Verweildauer und **polynomielle** Schlangengrößen, wenn die Injektionsrate r hinreichend klein ist?
  - Exponentiell große Verweildauer und Schlangengröße kann für hinreichend großes $r < 1$ nachgewiesen werden.
- Besitzt Longest-In-System eine **polynomielle** Verweildauer für alle $r < 1$?

  Es ist bekannt, dass die Verweildauer **exponentiell** im Durchmsser sein kann.
- Ist Longest-In-System für $r = 1$ stabil?