Praktikum Simulative Untersuchung von Protokollfunktionen des Internet

Aufgabenblatt 2 [1]

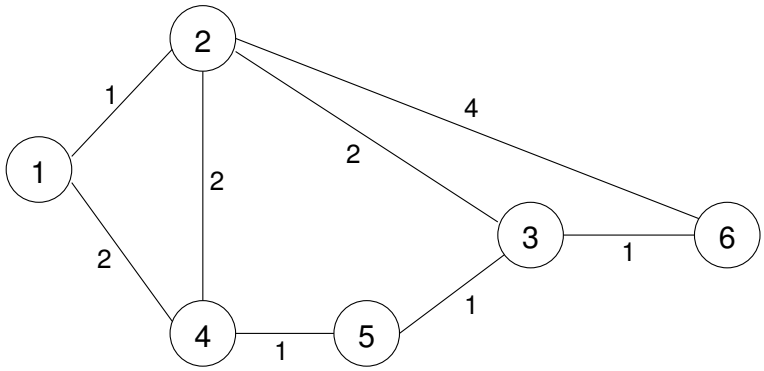
Vorbereitung

Aufgabe 1: Forwarding/Routing
Bei den Protokollfunktionen unterscheidet man unter anderem Forwarding und Routing. Erläutern Sie bitte den Unterschied zwischen den beiden Funktionen.

Aufgabe 2: Protokollfelder
Anhand welcher Protokollfelder kann der IP-Stack erkennen zu welchem Host und zu welcher Applikation auf diesem ein Paket zugestellt werden muss? Wie heißen die entsprechenden Felder im NetworkPacket des Protsim-Frameworks?

Aufgabe 3: Routing
Was ist eine Routing-Metrik? Was versteht man in diesem Zusammenhang unter einem kürzesten Pfad?

Aufgabe 4: Routing
Zeichnen Sie bitte in folgende Grafik den kürzesten Pfad von Knoten 1 zu Knoten 6 bei der gegebenen Metrik ein. Gehen Sie davon aus, dass diese Metrik symmetrisch ist, d.h. für beide Richtungen des Links gilt.



Praktischer Teil

Denken Sie bitte – wie immer bei den praktischen Aufgaben – an aussagekräftige Debug-Ausgaben.

Aufgabe 5: Forwarding
Im Verzeichnis `protsim02` liegt ein etwas umfangreicheres Programm als in der vorherigen Übung. Leider ist der Networkstack (`protsim02/common/NetworkStack.cc`) unvollständig:

1. In der Methode `getExternalGate()` muss das passende externe Gate zu dem übergebenen internen Gate ermittelt werden.
2. In `sendOut()` muss das passende Gate für den Zielknoten herausgefunden werden und dann die Nachricht entsprechend weiter gesendet oder (mit einer Fehlermeldung an das Environment) gedroppt (gelöscht) werden.

[1] Stand: 12. Oktober 2023

3. In der Funktion `deliverLocally()` muss das passende Gate für die Zielapplikation herausgefunden werden und dann die Nachricht entsprechend weiter gesendet oder (mit einer Fehlermeldung an das Environment) gedroppt (gelöscht) werden.

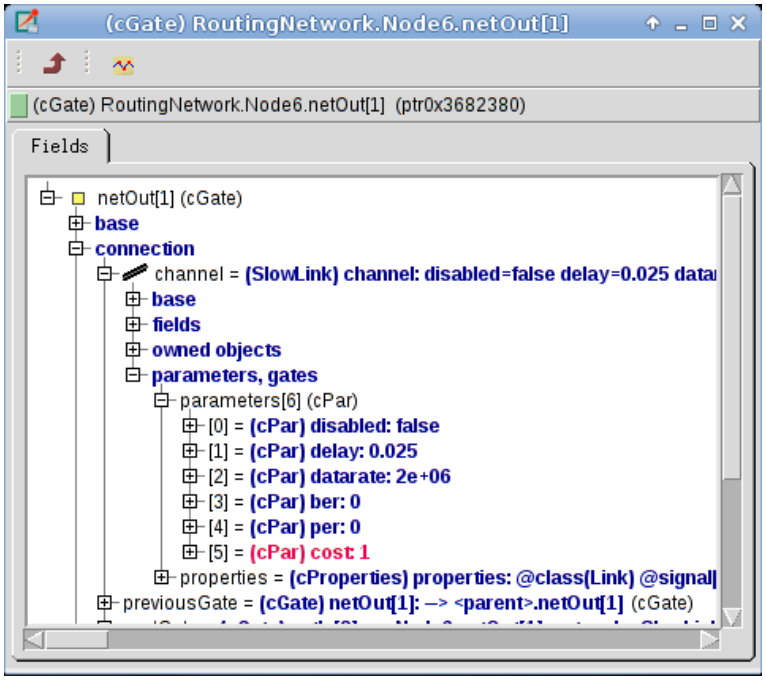
Finden Sie mittels Online-API-Dokumentation (siehe Hinweis weiter unten) heraus, wie Sie die Forwarding-Table bzw. die Tabelle der lokalen Applikationen auslesen können und ergänzen Sie bitte in der Datei `protsim02/common/NetworkStack.cc` die genannten Funktionen.

Achten Sie darauf, aussagekräftige Ausgaben zu erzeugen, um nachvollziehen zu können, wann welcher Node Nachrichten wohin forwarded (oder gar droppt). Kompilieren Sie die Simulation und führen Sie sie aus.

Aufgabe 6: Shortest-Path Routing
Neben dem Forwarding werden Sie übrigens auch das Routing vergeblich suchen, da Sie das in den folgenden Terminen noch selbst implementieren werden. Damit die Simulation in diesem Termin trotzdem funktioniert, haben wir das Routing-Modul in verschleierter Form für Sie dazugelegt.

Für diese Aufgabe soll die Metrik der Links geändert werden.

Um die Kosten eines Links zu ändern, müssen Sie in der Simulationsumgebung den Link auswählen (`Rechtsklick auf einen Link → Untermenu für einen der beiden gerichteten Links → open details`). Anschließend müsste sich ein ähnliches Dialogfenster öffnen:



Ändern Sie nun den `cost` Parameter. Die Linkkosten können nur im pausierten Zustand geändert werden. Anschließend sollte im Hauptfenster eine Meldung erscheinen, z.B. `Change link (Node6->Node2) costs from: 1 to: 2`.

Beachten Sie bitte, dass sich die Kosten jeweils nur auf den ausgehenden Link beziehen. Ein Link könnte durchaus in verschiedene Richtungen verschiedene Kosten haben und sie müssen diese (auch wenn sie gleich sind) getrennt setzen.

Experimentieren Sie mit den Link-Kosten und beobachten Sie, wie sich dies auf die Routen der Nachrichten auswirkt.

Ändern Sie nun bitte die Metrik der Links so, dass sie Aufgabe 4 entspricht und beobachten Sie, wie sich das auf das Routing auswirkt. Wo nehmen Nachrichten nun andere Wege? Können Sie sich das erklären?

Hinweise

Online-API-Dokumentation

Die Online-API-Dokumentation müssen Sie zunächst erstellen, wechseln Sie dazu in das Verzeichnis `protsim02/doc` und führen Sie den Befehl `doxygen` aus. Die Dokumentation finden Sie nun als Html Dokument im Ordner `api`. Öffnen Sie die Datei `api/index.html` mit einem Browser.

Debug Meldungen

Sie sollen in allen Protsim Aufgaben aussagekräftige Debug Meldungen ausgeben. Damit diese Meldungen auffällig und einfach zu finden sind, besitzt das Protsim Framework eine spezielle Log Klasse (`support/PSLog.h`). Sie bietet zum Loggen von Nachrichten einige Makros:

```
1 RED("Rote Nachricht");
2 GREEN("Nachricht");
3 YELLOW("Nachricht");
4 PURPLE("Nachricht");
5 BLUE("Nachricht");
6 CYAN("Nachricht");
7 LOG(loglevel, "Nachricht");
```

Im Terminal wird dann nicht nur die Nachricht, sondern auch der Dateiname und die Zeilennummer der Nachricht mit ausgegeben.