Matthias Simon, 1658438

Betriebssysteme, Übungsblatt 2, Winter 2025

Das zweite Übungsblatt beschäftigt sich mit dem „Santa Claus Problem“ zum Thema Synchronisation in Betriebssystemen. Ziel der Aufgabe ist die Koordination von drei Gruppen – dem Weihnachtsmann, den Rentieren und den Elfen – sicherzustellen.

Der Bericht gliedert sich in zwei Hauptteile: Die erste Aufgabe behandelt eine Implementierung mittels Semaphore in Python. Die zweite Aufgabe behandelt Docker-Container, die mittels ZeroMQ nachrichtenbasiert kommunizieren.

Aufgabe 1: Semaphore-basierte Implementierung

Zur Lösung des Problems wurde die Programmiersprache Python mit dem Modul *threading* verwendet. Das Programm besteht aus einem Weihnachtsmann-Thread, r Rentier-Threads und e Elfen-Threads, die parallel agieren. Bei der Implementierung gibt es folgende Bedingungen zu beachten:

- Die Rentier-Barriere: Der Weihnachtsmann darf erst geweckt werden, wenn alle r Rentier aus dem Urlaub zurückgekehrt sind.
- Gruppierung der Elfen: Um den Weihnachtsmann aufzusuchen müssen sich exakt p Elfen finden, die Hilfe benötigen. Weitere Elfen müssen warten, bis der Weihnachtsmann den p Elfen geholfen hat.

Die Synchronisation wird ausschließlich mit Hilfe von Semaphoren realisiert. C*ounterMutex* (Semaphore(1)) dient als binärer Semaphore, um die Variablen reindeer_counter und elf_counter vor Race Conditions zu schützen, bei denen Zählungen verloren gehen. Der Semaphore erzwingt, dass zu jedem Zeitpunkt nur ein Thread den Zähler schreibend verändern kann. *ElfMutex* (Semaphore(1)) stellt sicher, dass maximal p Elfen, in den Wartebereich eintreten können. Ist elf_counter < p wird der Semaphore sofort wieder freigegeben. Der Elf, der die Gruppe vervollständigt, gibt den Semaphore nicht direkt frei, sondern hält ihn im Zustand acquired. Die Blockade wird erst aufgehoben (release), wenn dem letzten Elf der Gruppe geholfen wurde. SantaSem (Semaphore(0)) steuert das Schlafen des Weihnachtsmanns. Lediglich das letzte ankommende Rentier und der letzte Elf einer Gruppe können den Weihnachtsmann mit release() aufwecken. ReindeerSem und ElfSem steuern das Verhalten der Rentiere und Elfen. Solange sie auf den Weihnachtsmann warten, sind sie im blockierten Zustand. Nachdem der Weihnachtsmann ihnen geholfen hat oder

Weihnachten vorbei ist, werden sie freigegeben. Kommen sie nun nach dem Urlaub am Nordpol an, oder benötigen erneut Hilfe, werden sie in den Zustand acquired versetzt.

Ablauflogik der Akteure

Der Weihnachtsmannthread läuft in einer Endlosschleife und befindet sich standardmäßig in einem Ruhezustand und wartet darauf von einem Elfen oder einem Rentier geweckt zu werden. Zunächst wird geprüft, ob er von einem Rentier geweckt wurde. Dies ist der Fall, wenn alle Rentiere am Nordpol sind. Sind alle Rentiere vor Ort, wird die Geschenkeauslieferung eingeleitet – selbst, wenn gleichzeitig genug Elfen auf Hilfe warten. Erst wenn die Rentierbedingung nicht erfüllt ist, wird geprüft, ob eine volle Elfengruppe Hilfe benötigt. Nach Beendigung der Tätigkeiten signalisiert der Weihnachtsmann den wartenden Threads über deren Semaphore, dass sie fortfahren können.

Die Rentiere kehren nach einer zufälligen Zeitspanne aus dem Urlaub zurück. Jedes zurückkehrende Rentier erhöht, geschützt durch den Mutex, den Zähler angekommener Rentiere. Das letzte Ankommende Rentier hat die Aufgabe den schlafenden Weihnachtsmann über santaSem.release() zu wecken. Anschließend warten alle Rentiere an einer Barriere (reindeerSem), bis die Auslieferung beendet ist und der Weihnachtsmann die Semaphore wieder freigibt.

Damit ein Elf seinen Hilfebedarf anmelden kann, muss er das Semaphor *elfMutex* passieren, das als Eintrittsschleuse agiert. Nachdem der Elf passieren konnte, erhöht er geschützt durch counterMutex die Anzahl wartender Elfen um eins. Solange die Gruppe noch unvollständig ist, gibt der Elf den elfMutex sofort wieder frei, damit weitere Elfen nachrücken können. Der Elf, der die Gruppe vervollständigt, blockiert der Mutex bewusst und weckt den Weihnachtsmann auf. Das verhindert, dass weitere Elfen in den Wartebereich eintreten können, während der Weihnachtsmann den Elfen hilft. Nachdem den Elfen geholfen wurde, werden sie wieder freigegeben. Der Elf, dem zuletzt geholfen wurde, gibt den elfMutex wieder frei und ermöglicht so die Bildung einer neuen Gruppe.

Abbildung 1 zeigt den korrekten Ablauf: Zunächst benötigen vier Elfen Hilfe (13, 12, 1, 3). Während der Weihnachtsmann diesen Elfen hilft, kehren die letzten Rentiere (4, 7, 8) zurück. Nachdem allen Elfen geholfen werden konnte, kann der Schlitten abflugbereit gemacht werden und die Geschenke ausgeliefert werden. Während die Geschenke ausgeliefert

werden, benötigen erneut 4 Elfen Hilfe. Diese Hilfe erhalten sie jedoch erste, nachdem der Weihnachtsmann vom Ausliefern der Geschenke zurückgekehrt ist.

```
Elf 13 benötigt Hilfe (1/4)
Elf 12 benötigt Hilfe (2/4)
Elf 1 benötigt Hilfe (3/4)
Elf 3 benötigt Hilfe (4/4)
4 Elfen brauchen Hilfe vom Weihnachtsmann.
Der Weihnachtsmann hilft den 4 Elfen
Rentier 4 ist zurück am Nordpol (8/10)
Rentier 7 ist zurück am Nordpol (9/10)
Rentier 8 ist zurück am Nordpol (10/10)
Allen Elfen wurde geholfen. Der Weihnachtsmann schläft wieder.
Alle Rentiere sind zurück. Schlitten kann abflugbereit gemacht werden
Der Schlitten ist abflugbereit und die Geschenke werden ausgeliefert
Elf 2 benötigt Hilfe (1/4)
Elf 6 benötigt Hilfe (2/4)
Elf 14 benötigt Hilfe (3/4)
Elf 15 benötigt Hilfe (4/4)
Der Weihnachtsmann ist zurück vom Ausliefern der Geschenke. Die Rentiere können wieder in den Süden fliegen. Der Weihnachtsmann schläft wieder.
4 Elfen brauchen Hilfe vom Weihnachtsmann.
Der Weihnachtsmann hilft den 4 Elfen
Allen Elfen wurde geholfen. Der Weihnachtsmann schläft wieder.
Elf 5 benötigt Hilfe (1/4)
Elf 4 benötigt Hilfe (2/4)
Elf 8 benötigt Hilfe (3/4)
Elf 10 benötigt Hilfe (4/4)
4 Elfen brauchen Hilfe vom Weihnachtsmann.
Der Weihnachtsmann hilft den 4 Elfen
```

Abbildung 1: Protokoll der Semaphor-basierten Implementierung

Aufgabe 2: Docker-basierte Implementierung

Im zweiten Teil der Übung läuft nun jeder Akteur (Weihnachtsmann, Rentier, Elf) in einem eigenen Docker-Container. Da in dieser Architektur kein gemeinsamer Speicher existiert, wurde die Synchronisation auf Nachrichtenaustausch mittels ZeroMQ umgestellt.

Die Architektur folgt einem Client-Server-Modell. Der Weihnachtsmann-Container fungiert als Server, der den globalen Systemzustand (Anzahl wartender Elfen und Rentiere) verwaltet und Entscheidungen trifft. Die Elfen- und Rentier-Container agieren als Clients, die Statusänderungen an den Server melden und anschließend blockierend auf dessen Anweisungen warten. Alle Container kommunizieren über ein Docker-Bridge-Netzwerk (northpole_net). Der Weihnachtsmann ist unter dem Hostnamen „santa“ und Port 5555 erreichbar. Für die Vernetzung der Container wurde das asynchrone ROUTER-DEALER-Pattern von ZeroMQ verwendet. Im Gegensatz zum klassischen Request-Reply-Muster (REQ-REP), das eine sofortige Antwort erzwingt, erlaubt diese Konstellation eine zeitliche Entkopplung von Anfrage und Antwort. Wenn ein Elf eine Anfrage sendet, kann der Weihnachtsmann (Router) nicht sofort antworten, sondern muss warten, bis sich eine vollständige Gruppe

gebildet hat. Der Weihnachtsmann speichert die ID des Elfen zwischen und sendet die Antwort erst zu einem späteren Zeitpunkt zurück.

Generierung der Docker-Container

Da das manuelle Erstellen einer docker-compose.yml für 1+e+r Container fehleranfällig und unflexibel ist, wurde ein Python-Skript entwickelt. Zunächst wird festgelegt, wie viele Elfen und Rentiere generiert werden sollen. Anschließend wird über diese Anzahl iteriert, wobei jedem Rentier und jedem Elfen eine eindeutiger Containername verteilt wird. Beim Starten der Elfen-container wird „command: python -u elf/main.py --id {i}“ aufgerufen. Damit wird der Python-Interpreter gestartet. Die Verwendung des Flags -u (unbuffered) sorgt dafür, dass die Ausgabe nicht gepuffert wird. Ansonsten könnten die Logs verzögert oder gar nicht erscheinen. Das aufzurufende Python-Skript liegt im Ordner elf mit dem Namen main.py. Zudem wird beim Aufruf die ID des Elfen mit übergeben. Das Starten der Rentier-Container erfolgt ähnlich.

Ablauflogik der Akteure

Der Container des Weihnachtsmanns steuert das gesamte Verhalten aller Akteure. Implementiert in einer Ereignisschleife wartet er permanent auf eingehende Nachrichten am Router-Socket. Trifft eine Nachricht ein, enthält diese die Identität des Senders in Form der ID sowie den Nachrichteninhalt. Der Nachrichteninhalt kann dabei entweder I_NEED_HELP von einem Elfen oder I_AM_HERE von einem Rentier sein. Basierend darauf wird die Anzahl wartender Elfen oder angekommener Rentiere aktualisiert. Nach dieser Aktualisierung wird geprüft, ob alle Rentiere angekommen sind, oder ob genug Elfen Hilfe benötigen. Ist eine der Bedingungen erfüllt sendet der Weihnachtsmann eine Nachricht an die betroffenen Clients und geht in eine simulierte Arbeitsphase. Nach getaner Arbeit bekommen die Rentiere die Nachricht wieder in den Süden zu fliegen oder die Elfen die Nachricht wieder an den Geschenken zu arbeiten.

Die Elfen folgen einem Zyklus aus vier Schritten. Zunächst arbeitet der Elf an den Geschenken. Nach einer gewissen Zeit tritt ein Problem auf und der Elf sendet eine Nachricht an den Server. Unmittelbar nachdem Senden tritt der Elf in eine innere Empfangsschleife ein (socket.rev()) und wartet auf eine Antwort. Sobald genug Elfen Hilfe benötigen, sendet der Weihnachtsmann zunächst HELPING_YOU an alle hilfesuchenden Elfen und nach dem Abschluss der Hilfe sendet er BACK_WORK, womit der Zyklus von vorne beginnt.

Die Rentiere folgen einem ähnlichen Muster. Das Rentier verbringt eine gewissen Zeit im Süden (simuliert durch time.sleep). Nach Ankunft am Nordpol sendet es I_AM_HERE an den Server. Das Rentier ruft dann socket.recv() auf und blockiert. Es verbleibt in diesem Zustand, bis alle anderen Rentiere eingetroffen sind. Sobald der Weihnachtsmann den Schlittenflug imitiert, erhalten die Rentiere die Nachricht DELIVERING_GIFTS und anschließend FLY_SOUTH, womit die Blockade gelöst wird und der Zyklus wieder von vorne beginnt.

Die Validierung erfolgt anhand der generierten Log-Ausgabe (siehe Abbildung 2). Die Analyse zeigt das korrekte Verhalten des Programms:

1. **Nebenläufigkeit:** Zu Beginn des Logs ist erkennbar, dass Nachrichten von Elfen und Rentieren zeitlich vermischt beim Server eintreffen. Der Server verarbeitet diese Anfragen sequenziell, was durch die inkrementell steigenden Zählerstände (z. B. Elfen 1/4 bis 4/4) belegt wird.
2. **Entscheidung an der Grenzschwelle:** Während bereits neun Rentiere warten, trifft in diesem Moment ein vierter Elf ein, der eine Gruppe vervollständigt. Da die Rentier-Bedingung (< 10) noch nicht erfüllt ist, entscheidet der Algorithmus, zunächst den Elfen zu helfen. Dies vermeidet unnötige Wartezeiten der Elfen.
3. **Priorisierung:** Unmittelbar nach der Hilfe für die Elfen trifft das letzte fehlende Rentier ein. Der Server löst sofort den Zustandswechsel zur Geschenkauslieferung aus. Ab diesem Moment werden Elfenanfragen zurückgestellt.
4. **Reset:** Nach Rückkehr der Rentiere in den Süden leeren sich die Listen, und der Weihnachtsmann ist bereit für den nächsten Zyklus.

Abbildung 3 zeigt einen Sonderfall. Dort wartet ein Elf auf die Hilfe des Weihnachtsmanns, als das zehnte Rentier zurückkommt. Der Prozess des Geschenkeausliefern wird in Gang gesetzt. Erst nachdem der Weihnachtsmann zurückgekehrt ist, können Elfen wieder um Rat fragen, wobei die Anzahl wartender Elfen korrekt gespeichert wird.

```
santa  | Elf Elf-8 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Elf Elf-15 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Elf Elf-1 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Rentier Rentier-8 ist zurück. Insgesamt zurückgekehrte Rentiere: 4/10
santa  | Rentier Rentier-4 ist zurück. Insgesamt zurückgekehrte Rentiere: 5/10
santa  | Elf Elf-2 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Elf Elf-5 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Rentier Rentier-5 ist zurück. Insgesamt zurückgekehrte Rentiere: 6/10
santa  | Elf Elf-7 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Rentier Rentier-10 ist zurück. Insgesamt zurückgekehrte Rentiere: 7/10
santa  | Elf Elf-6 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Elf Elf-10 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Rentier Rentier-3 ist zurück. Insgesamt zurückgekehrte Rentiere: 8/10
santa  | Elf Elf-14 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Elf Elf-3 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Rentier Rentier-1 ist zurück. Insgesamt zurückgekehrte Rentiere: 9/10
santa  | Elf Elf-8 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Rentier Rentier-6 ist zurück. Insgesamt zurückgekehrte Rentiere: 10/10
santa  | Alle Rentiere sind da. Schlitten kann abflugbereit gemacht werden.
santa  | Geschenke wurden geliefert. Rentiere fliegen zurück in den Süden.
santa  | Elf Elf-13 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Elf Elf-12 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Elf Elf-15 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Elf Elf-4 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Elf Elf-5 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Elf Elf-1 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Elf Elf-7 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Elf Elf-2 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Elf Elf-9 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Elf Elf-11 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Rentier Rentier-4 ist zurück. Insgesamt zurückgekehrte Rentiere: 1/10
santa  | Elf Elf-6 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Elf Elf-10 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
santa  | Rentier Rentier-3 ist zurück. Insgesamt zurückgekehrte Rentiere: 2/10
santa  | Elf Elf-14 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Elf Elf-3 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Rentier Rentier-5 ist zurück. Insgesamt zurückgekehrte Rentiere: 3/10
```

Abbildung 2: Protokoll der Container-basierten Implementierung

```
santa  | Elf Elf-1 braucht Hilfe. Insgesamt wartende Elfen: 1/4
santa  | Rentier Rentier-5 ist zurück. Insgesamt zurückgekehrte Rentiere: 10/10
santa  | Alle Rentiere sind da. Schlitten kann abflugbereit gemacht werden.
santa  | Geschenke wurden geliefert. Rentiere fliegen zurück in den Süden.
santa  | Elf Elf-5 braucht Hilfe. Insgesamt wartende Elfen: 2/4
santa  | Elf Elf-10 braucht Hilfe. Insgesamt wartende Elfen: 3/4
santa  | Elf Elf-15 braucht Hilfe. Insgesamt wartende Elfen: 4/4
santa  | Genug Elfen sind da. Ich helfe ihnen jetzt.
santa  | Allen Elfen wurde geholfen.
```

Abbildung 3