

Aufgabe: Der größte gemeinsame Teiler

In dieser Aufgabe ist ein **verteilter Algorithmus** zu implementieren. Jeder Prozess (oder Thread) durchläuft den gleichen Algorithmus!

Hintergrundwissen

Verwendet wird einer der ältesten, bekannten Algorithmen: Euklids Algorithmus zur Bestimmung des ggT, beschrieben im 7. Buch der Elemente um ca. 300 v.Chr.

Satz von Euklid: Der größte gemeinsame Teiler (ggT) zweier positiver ganzer Zahlen x, y (mit $x \geq y > 0$) ist gleich dem ggT von y und dem Rest, der bei ganzzahliger Division von x durch y entsteht.

Die Formalisierung des Originals lautet etwa wie folgt:

```
ggT (x;y) // requires  $x \geq y > 0$ , ensures return  $> 0$ 
  if  $x = y$ 
    then return  $x$  //ggT( $x, x$ ) =  $x$ 
    else if  $x > y$ 
      then ggT (y;x)
      else ggT (x;  $y - x$ ) //ggT( $x, y$ ) = ggT( $y, \text{mod}(x, y)$ )
  fi fi fi
```

Im von Euklid angegebenen Algorithmus wird kein mod-Operator verwendet! Um die Abfrage auf Gleichheit einzusparen und dennoch zu einem korrekten Rekursionsabbruch zu gelangen, wird der mod-Aufruf leicht verändert: $\text{mod}^*(x,y) := \text{mod}(x-1,y)+1$ (Zeile 3 im folgenden Algorithmus).

Verteilter Algorithmus

1. Jeder Prozess P_i (oder als Thread realisiert) hat seine eigene Variable M_i mit Wert $\#M_i$
2. ggT aller am Anfang bestehender $\#M_i$ wird berechnet:
{Eine Nachricht mit der Zahl $\#y$ ist eingetroffen}
if $\#y < \#M_i$
 then $M_i := \text{mod}(\#M_i - 1, \#y) + 1$; // Berechnung von $\text{mod}^*(x,y)$
 send $\#M_i$ to all neighbours; // Rekursionsaufruf
fi

Aufgabenstellung

Die aufgeführten Aufgaben stellen nicht die Implementierungsreihenfolge dar!

- Implementieren Sie einen **Koordinator**, der von den (im Folgenden noch beschriebenen) Startern die Identitäten (z.B. Port + Rechner) der Prozesse P_i erhält und diese in einem logischen Ring anordnet. Der Ring soll so aufgebaut werden, das benachbarte Prozesse P_i und P_{i+1} nach Möglichkeit auf verschiedenen Rechnern laufen. Dazu sendet er allen Prozessen P_i die Identität und Position (links oder rechts) ihrer Nachbarn. Implementieren Sie eine Bedienungsoberfläche für den Koordinator, über die ein Anwender die Anzahl der Prozesse, den gewünschten ggT und ein Intervall für die Berechnungsdauer eingeben kann.
- Implementieren Sie einen **Starter**, der auf seinem Rechner eine bestimmte Anzahl an Prozessen P_i startet. Die Anzahl ist bei dem Koordinator zu erfragen. Jeder Prozess erhält eine Zahl, die sich durch Zufallszahl * gewünschter-ggT berechnet. Der gewünschte ggT ist bei dem Koordinator zu erfragen. Zudem erhält jeder Prozess als Berechnungszeit eine Zufallszahl aus dem bei dem Koordinator zu erfragenden Berechnungsintervall.

- Implementieren Sie die Prozesse P_i : Die Prozesse müssen im Zustand "Der Ring wird aufgebaut" ihre Zahl M_i und die beiden Nachbarn empfangen und sich merken können. Zudem ist die Berechnung im Falle einer Neuberechnung der eigenen Zahl exakt in der durch den Starter vorgegebenen Berechnungszeit durchzuführen, d.h. ggf. ist sie zu verzögern.
- Implementieren Sie einen **Monitor**, der einen Überblick über den aktuellen Systemzustand liefert.
- Implementieren Sie ein Verfahren **StartBerechnung**, um die Berechnung zu starten. Der für den Start gewählte Algorithmus ist im Monitor nachvollziehbar anzuzeigen, z.B. indem die am Start beteiligten Prozesse P_i im Monitor kenntlich gemacht werden.
- Implementieren Sie ein Verfahren **TerminiereBerechnung**, um die Berechnung zu beenden. Der für die Beendung gewählte Algorithmus ist im Monitor nachvollziehbar anzuzeigen.
- Die von dem System als ggT bestimmte Zahl ist im Monitor nach Terminierung anzuzeigen.

Freiheitsgrade

In der Aufgabenstellung sind bestimmte Punkte noch nicht festgelegt worden, die für die Lösung festgelegt werden müssen. Die aufgeführten Prozesse sind entsprechend der gewählten Lösung zu erweitern.

- Zur **Kommunikation** kann UDP oder TCP verwendet werden, aber auch DCOM, CORBA, Java-RMI, .NET, SOAP etc. Die Wahl ist zu begründen und ein mögliches Systemumfeld dazu zu beschreiben. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen.
- Der Algorithmus zum **Start** der Berechnung ist nicht festgelegt worden. Hier kann von einer dezentralen Lösung, z.B. per Wahl, über eine randomisierte Lösung bis zu einer zentralen Lösung alles verwendet werden. Die Wahl ist zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen. Insbesondere ist zu klären, ob der Algorithmus bei dem gewählten Verfahren immer terminiert bzw. wann er nicht terminiert.
- Der Algorithmus zum **Beenden** des Verfahrens ist nicht festgelegt worden. Auch hier kann von einer dezentralen Lösung, z.B. per Wahl, über eine randomisierte Lösung bis zu einer zentralen Lösung alles verwendet werden. Die Wahl ist zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen. Insbesondere ist zu klären, ob der Algorithmus bei dem gewählten Verfahren immer terminiert bzw. wann er nicht terminiert und ob der Algorithmus immer korrekt arbeitet, d.h. alle Prozesse den gleichen ggT bestimmt haben.
- Es ist nicht festgelegt worden, was als **aktueller Systemzustand** im Monitor angezeigt werden soll. Die Bestandteile des Systemzustands, wie etwa die aktuellen Werte der Prozesse oder auch die gerade im Umlauf befindlichen Nachrichten, evtl. inklusive der übertragenen Werte, kann frei, aber sinnvoll bestimmt werden. Zudem kann zwischen gesichertem globalem Systemzustand und möglicherweise bestehendem Systemzustand gewählt werden. Auch hier ist die Wahl zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen.
- Die **Fehlersemantik** ist nicht festgelegt worden: Was passiert bei Nachrichtenverlusten oder Ausfall von Prozessen etc. Hier kann eine beliebige Fehlersemantik gewählt werden. Die Wahl ist zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen. Um Ihr eigenes Vertrauen in Ihr Programm zu dokumentieren, ist ein Betrag in € anzugeben, den Sie im Falle eines nicht korrekten Terminierens (also keine korrekte Berechnung des ggT) als Schadensleistung zahlen würden.