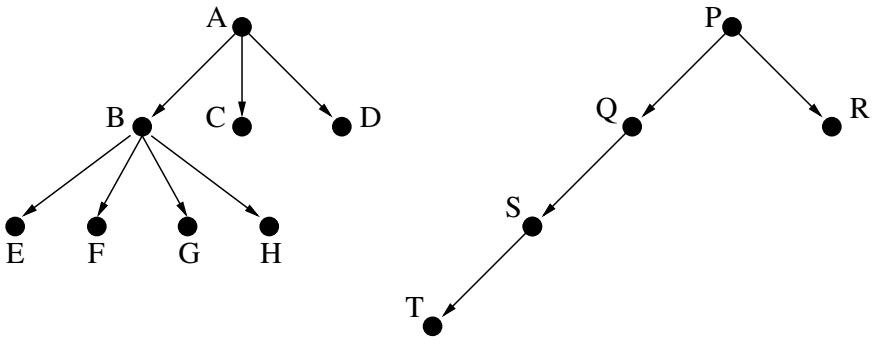
– Praktikumsaufgabe 5 –

Thema: *Scheduling*

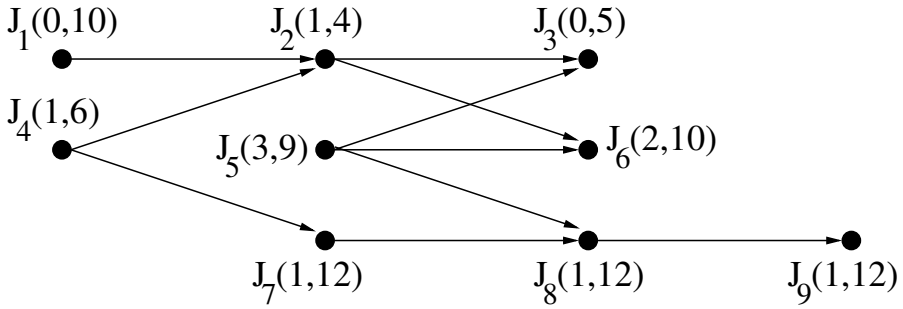
Zielstellung: Verständnis des Präzedenzgraphen (1. ,2.), Einfache Schedulingverfahren: EDF (2.), LST (3.) Verstehen der Nutzung von Network Flows für die Ermittlung zeitgesteuerter Schedules (4. ,5)

1. Gegeben sei der Präzedenzgraph einer Gruppe von Jobs; die Ausführungszeit betrage einheitlich $t_e = 1$, alle Jobs seien ab $t_r = 0$ bereit.



   Ermitteln Sie einen Schedule für 3 Prozessoren, der die mittlere Komplettierungszeit t_c minimiert! (Es ist irrelevant, ob präemptiv oder kooperativ gearbeitet wird, da jeder Job sowieso nur 1 Zeiteinheit benötigt.) Beschreiben Sie den genutzten Algorithmus!

2. Gegeben sei der Präzedenzgraph mit $J_x(t_r, t_d)$, die Ausführungszeit jedes Jobs betrage $t_e = 1$.



   a) Ermitteln Sie die effektiven Bereitzeiten und Deadlines jedes Jobs!

   b) Generieren Sie einen gültigen Schedule nach EDF für einen Prozessor!

   c) Jeder Job habe eine Priorität i, die durch den *längsten* Pfad bis zu einem Job ohne Nachfolger bestimmt wird (z.B. hat J_9 die Priorität 0, und J_5 die Priorität 2). Je kleiner i desto kleiner die Priorität. Ermitteln Sie einen entsprechenden prioritätsgesteuerten Schedule für einen Prozessor!

3. Gegeben seien 5 periodische Tasks A, B, C, D, E und 3 Prozessoren. Die Periode von A, B und C sei $t_p = 2$, und ihre Ausführungszeit sei $t_e = 1$. Die Parameter von D und E sind $t_p = 8$ und $t_e = 6$. Die (relativen) Deadlines sind identisch zu den Perioden.

   a) Zeigen Sie, dass nicht alle Jobs ihre Deadlines einhalten, wenn die Menge dynamisch nach LST geschedult wird!

   b) Ermitteln Sie einen brauchbaren Schedule!

4. Implementieren Sie ein Programm, welches zu einer Taskbeschreibung in Textform geeignete Framegrößen errechnet. Ermitteln Sie damit für die folgenden Taskmengen geeignete Framegrößen!

   a) (6,1), (10,2) und (18,2)

   b) (8,1), (15,3), (20,4) und (22,6)

   Zum Einlesen der Taskbeschreibung können Sie den auf der Website bereitgestellten Code (oder eine bessere Variante) nutzen. Falls Sie den bereitgestellten Code nutzen, achten Sie bitte darauf, `stdlib.h` zu inkludieren, anderenfalls liefert `strtod()` stillschweigend falsche Ergebnisse.

   Hinweis: Eine Schwierigkeit besteht in der Ermittlung von ggT (für das dritte Frame Size Constraint) und kgV (für die Hyperperiode), für die es keine einfachen Bibliotheksimplementierungen gibt. Den ggT können Sie z. B. einfach rekursiv mit dem euklidischen Algorithmus implementieren:

```
unsigned long gcd_euclid(unsigned long a, unsigned long b)
{
    if (b==0) {
        return a;
    }
    else {
        return gcd_euclid(b, (a % b)) ;
    }
}
```

   Das kgV zweier Operanden kann dann mittels aus dem ggT folgendermaßen ermittelt werden:

$$\mathrm{kgV}(a,b) = \frac{a}{\mathrm{ggT}(a,b) \cdot b}$$

5. Gegeben sind die beiden Tasks $T_i(t_{p,i}, t_{e,i})$ $T_1(4,3)$ und $T_2(6,1.5)$.

   a) Ermitteln Sie geeignete Größen für die Frames!

   b) Zeichnen Sie den Network Flow Graph und leiten Sie daraus einen Schedule ab!

   (Quelle: Jane Liu: *Real-Time Systems*. Prentice Hall, 2000, S. 114, Aufgabe 5.1 a) und b))