For the finiteness result, we define $u = \sum_{i=1}^{n}(e_i/P_i)$, $d_{\max} = \max_{1 \le i \le n}\{d_i\}$ and $P = \text{lcm}(P_1, \ldots P_n)$. Define $h_T(t)$ to be the sum of the execution times of all the tasks in set $T$ whose absolute deadlines are less than or equal to $t$.

**Theorem 3.12.** A task set of $n$ tasks is not EDF-feasible iff

- $u > 1$ or
- there exists

$$t < \min\left\{P + d_{\max}, \quad \frac{u}{1-u}\max_{1 \le i \le n}\{P_i - d_i\}\right\}$$

such that $h_T(t) > t$.

Under this theorem, we only need to check for feasibility up to some finite time. We can build the proof of Theorem 3.12 using the following series of lemmas.

**Lemma 3.9.** A given set $T$ of periodic tasks is not EDF-schedulable iff there exists some time $t$ such that $h_T(t) > t$.

***Proof.*** This has been left to the reader.

**Lemma 3.10.** Given a set $T$ of $n$ periodic tasks, if $u \le 1$,

$$h_T(t+P) > t+P \Rightarrow h_T(t) > t \qquad \text{for all} \quad t \ge d_{\max}$$

***Proof***

$$\begin{aligned} h_T(t) + P &= \sum_{i=1}^{n} e_i\left(\left\lfloor\frac{t-d_i}{P_i}\right\rfloor + 1\right) + P \\ &\ge \sum_{i=1}^{n} e_i\left(\left\lfloor\frac{t-d_i}{P_i}\right\rfloor + 1\right) + P\sum_{i=1}^{n}\frac{e_i}{P_i} \\ &= \sum_{i=1}^{n} e_i\left(\left\lfloor\frac{t-d_i+P}{P_i}\right\rfloor + 1\right) \qquad \text{since } P \text{ is a multiple of } P_i \\ &= h_T(t+P) \end{aligned}$$

Hence,

$$h_T(t+P) > t+P \Rightarrow h_T(t) > t \tag{3.54}$$

**Q.E.D.**

**Lemma 3.11.** If task set $T$ is not EDF-feasible and $u \le 1$, then there exists $t < P + d_{\max}$ such that $h_T(t) > t$.

***Proof.*** Follows immediately from Lemma 3.10. **Q.E.D.**

**Lemma 3.12.** Suppose $T$ is not feasible and $u \leq 1$. Then $h_T(t) > t$ implies

$$t < d_{\max} \qquad \text{or} \qquad t < \max_{1 \leq i \leq n}\{P_i - d_i\}\frac{u}{1-u}$$

***Proof.*** Suppose that $t > d_{\max}$. We have

$$\begin{aligned} h_T(t) &\leq \sum_{i=1}^{n} e_i \frac{t - d_i + P_i}{P_i} \\ &= t\sum_{i=1}^{n}\frac{e_i}{P_i} + \sum_{i=1}^{n}\frac{P_i - d_i}{P_i} \\ &\leq \sum_{i=1}^{n}\left[\frac{e_i}{P_i}\left(t + \max_{1 \leq i \leq n}\{P_i - d_i\}\right)\right] \end{aligned} \tag{3.55}$$

If $h_T(t) > t$, we will have from (3.55),

$$\begin{aligned} t &< \sum_{i=1}^{n}\frac{e_i}{P_i}\left(t + \max_{1 \leq i \leq n}\{P_i - d_i\}\right) \\ \Rightarrow t &< \max_{1 \leq i \leq n}\{P_i - d_i\}\frac{u}{1-u} \end{aligned} \tag{3.56}$$

**Q.E.D.**

### 3.2.3 Allowing for Precedence and Exclusion Conditions*

We have assumed in the above sections that tasks are independent and are always preemptible by other tasks. We will now relax both these assumptions and present several scheduling heuristics.

Consider a set of tasks with a precedence graph, which are released at time 0. A deadline is specified for each task. It is assumed that the deadlines are chosen so that even if a task completes at its deadline, there will be enough time to execute its children in the task graph by their deadlines. If all the tasks that form a task graph are assigned to the same processor, then we can use the algorithm in Figure 3.19.

**Example 3.21.** Consider the task graph shown in Figure 3.20a, where the task execution times and deadlines are as follows:

| Task $T_i$ | $e_i$ | $d_i$ | Task $T_i$ | $e_i$ | $D_i$ |
|---|---|---|---|---|---|
| 1 | 3 | 6 | 2 | 3 | 7 |
| 3 | 2 | 20 | 4 | 5 | 21 |
| 5 | 6 | 27 | 6 | 6 | 28 |