Project Scheduling: CPM time analysis

1. Intoduzione

- Una volta costruita la rete di progetto AOA (attività sugli archi) $\mathcal{G}=(\mathcal{N}, \mathcal{A})$ è possibile fare una prima analisi del progetto a partire dalla conoscenza delle durate (tempi di esecuzione) t_{ij} di ogni attività (i, j).

- Le durate t_{ij} possono essere note con certezza come nel *CPM* oppure essere stocastiche e rappresentabili mediante variabili aleatorie come nel *PERT*.

- In ogni caso, in tutti i progetti esiste un *insieme di attività* che sono particolarmente *importanti* nella determinazione del tempo di completamento del progetto nel senso che un qualsiasi ritardo nel completamento di una di esse si ripercuote integralmente sul tempo di esecuzione complessivo.

- Queste attività sono le cosiddette *attività critiche* di un progetto

Project Scheduling: CPM time analysis

2. Percorso critico

- Sia T il *tempo minimo di esecuzione di un progetto* in corrispondenza a certi prefissati valori $\{t_{ij}\}$ dei tempi di esecuzione delle singole attività.

Definizione 1: Attività critica

- Un'*attività* (i, j) si dice "*critica*" se sostituendo t_{ij} con $t_{ij} + \Delta t$ il minimo di tempo di esecuzione del progetto diventa $T + \Delta t$.

Definizione 2: Evento critico

- Un evento i si dice "*critico*" se posticipando di Δt il suo verificarsi rispetto al minimo istante in cui può accadere il minimo tempo di esecuzione del progetto diventa $T + \Delta t$.

Definizione 3: Percorso critico

- Data una rete rappresentativa di un progetto si dice *percorso critico* qualsiasi cammino dal nodo iniziale al nodo finale in cui tutti gli archi siano rappresentativi di attività critiche.
- Un *percorso critico* è formato da soli *nodi* (eventi) *critici*, ma la *condizione* è *solo necessaria*.

Project Scheduling: CPM time analysis

3. Tempi di raggiungimento

- Per determinare il percorso critico di un progetto occorre introdurre altre due definizioni:

Definizione 4: Tempo minimo di raggiungimento

- Si definisce *tempo minimo di raggiungimento* t_i *di un evento* i il *minimo istante di tempo* entro cui *possono* essere completate *tutte* le attività che *precedono* l'evento i.

Definizione 5: Tempo massimo di raggiungimento

- Si definisce *tempo massimo di raggiungimento* T_i *di un evento* i il *massimo istante di tempo* entro cui *almeno* un'attività che *segue* l'evento i *deve cominciare*, pena un aumento del tempo minimo di esecuzione del progetto.

Osservazione

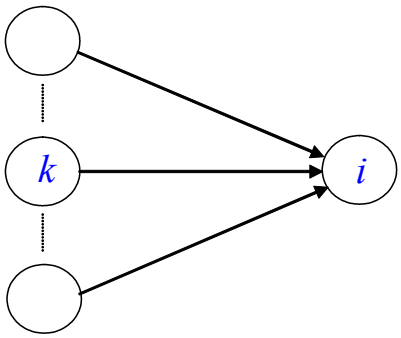
- Il tempo minimo T di esecuzione di un progetto coincide col tempo minimo di raggiungimento dell'evento finale f; risulta perciò che $T = t_f$.

Project Scheduling: CPM time analysis

3. Tempi di raggiungimento (continua)

Calcolo di t_i e T_i

(i) Supponiamo *noti i tempi* t_k *di tutti gli eventi* k *predecessori dell'evento* i.



La generica attività (k, i) che precede l'evento i può essere completata al più presto all'istante $t_k + t_{ki}$.

Pertanto, per definizione, $t_i \geq t_k + t_{ki}$.

Poiché in particolare t_i è il *minimo istante di tempo entro cui tutte le attività* (k, i) *possono essere completate*, allora si ha

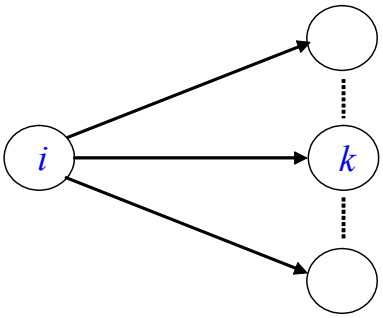
$$t_i = \max_{\{k\,:\,(k,\,i)\in\mathcal{A}\}} \{t_k + t_{ki}\},$$

cioè t_i è pari al *massimo* tra i *minimi tempi di completamento delle attività precedenti l'evento* i.

Project Scheduling: CPM time analysis

3. Tempi di raggiungimento (continua)

(ii) Supponiamo ora *noti i tempi T_k di tutti gli eventi k successori dell'evento i*.



La generica attività (i, k) che segue l'evento i deve essere completata entro l'istante T_k (pena un aumento del tempo minimo di esecuzione del progetto), e quindi deve avere inizio entro l'istante $T_k - t_{ik}$.

Pertanto, per definizione, $T_i \leq T_k - t_{ik}$.

Poiché in particolare T_i è il *massimo istante di tempo entro cui almeno un'attività (i, k) che segue l'evento i deve iniziare* (pena un aumento del tempo minimo di esecuzione del progetto), allora si ha:

$$T_i = \min_{\{k\,:\,(i,k)\in\mathcal{A}\}} \{T_k - t_{ik}\},$$

cioè T_i è pari al *minimo* tra i *massimi istanti di inizio delle attività successive all'evento i* (compatibilmente con la minima durata del progetto).

Project Scheduling: CPM time analysis

3. Tempi di raggiungimento (continua)

Algoritmo di calcolo

- Le precedenti definizioni consentono di formalizzare subito un *algoritmo in due fasi*. Sia $\mathcal{A}$ l'insieme degli archi del grafo di progetto.

(i) *Fase di calcolo in avanti*

$t_0 := 0$.
for $i := 1$ to f do
$t_i := \max_{\{k\,:\,(k,\,i)\in\mathcal{A}\}} \{t_k + t_{ki}\}$

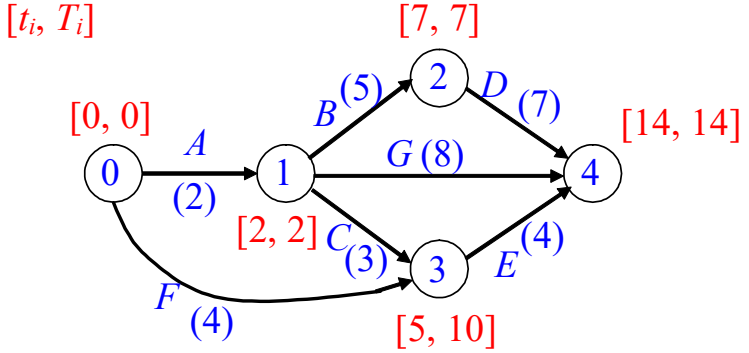
(ii) *Fase di calcolo all'indietro*

$T_f := t_f$.
for $i := f - 1$ down to 0 do
$T_i := \min_{\{k\,:\,(i,\,k)\in\mathcal{A}\}} \{T_k - t_{ik}\}$

Complessità delle due fasi $O(m)$, dove m è il numero di archi della rete di progetto.

Esempio

Attività: A, B, C, D, E, F, G
Durate: 2, 5, 3, 7, 4, 4, 8
Precedenze: $A < B$; $A < C$; $B < D$; $C < E$; $F < E$; $A < G$



Project Scheduling: CPM time analysis

4. Tempi minimi e massimi di inizio e completamento di un'attività

Definizione 6: *Tempo minimo di inizio di un'attività*

- Si definisce *tempo minimo di inizio* ES_{ij} *di un'attività* (i, j) il *minimo tempo* entro cui questa può iniziare: $ES_{ij} = t_i$.

Definizione 7: *Tempo minimo di completamento di un'attività*

- Si definisce *tempo minimo di completamento (fine)* EF_{ij} *di un'attività* (i, j) il *minimo tempo* entro cui questa può terminare: $EF_{ij} = ES_{ij} + t_{ij} = t_i + t_{ij}$.

Definizione 8: *Tempo massimo di completamento di un'attività*

- Si definisce *tempo massimo di completamento* LF_{ij} *di un'attività* (i, j) il *massimo tempo* entro cui questa deve essere completata (pena un aumento del tempo minimo di esecuzione del progetto): $LF_{ij} = T_j$.

Definizione 9: *Tempo massimo di inizio di un'attività*

- Si definisce *tempo massimo di inizio* LS_{ij} *di un'attività* (i, j) il *massimo tempo* entro cui questa deve iniziare (pena un aumento del tempo minimo di esecuzione del progetto): $LS_{ij} = LF_{ij} - t_{ij} = T_j - t_{ij}$.

Project Scheduling: CPM time analysis

5. Tempi di slittamento

Definizione 10: Slittamento di un evento

- Il *tempo di slittamento* F_i (o *margine di tempo*) dell'evento i è il *massimo ritardo* possibile sulla sua occorrenza *senza alterare il tempo minimo di completamento del progetto*: $F_i = T_i - t_i$.

Definizione 11: Evento critico

- L'*evento* i per cui $F_i = T_i - t_i = 0$ è *critico*.

Definizione 12: Slittamento totale di un'attività

- Lo *slittamento totale* TF_{ij} (o *margine di tempo*) dell'attività (i, j) è il *massimo ritardo* possibile sul suo inizio (completamento) *senza alterare il tempo minimo di completamento del progetto*:

$$TF_{ij} = LS_{ij} - ES_{ij} = (T_j - t_{ij}) - t_i =$$
$$= LF_{ij} - EF_{ij} = T_j - (t_i + t_{ij}).$$

Definizione 13: Attività critica

- L'*attività* (i, j) per cui $TF_{ij} = 0$ è *critica*.

Project Scheduling: CPM time analysis

5. Tempi di slittamento (continua)

Definizione 14: Slittamento libero di un'attività

- Lo *slittamento libero* FF_{ij} dell'attività (i, j) è il *massimo ritardo* possibile sul suo completamento *senza alterare il tempo minimo di inizio delle attività successive*:

$$FF_{ij} = \min_{\{k\,:\,(j,\,k)\in\mathcal{A}\}}\{ES_{jk}\} - EF_{ij} = \\ = t_j - (t_i + t_{ij}).$$

Definizione 15: Slittamento di sicurezza di un'attività

- Il *tempo di slittamento di sicurezza* SF_{ij} (o *margine di tempo di sicurezza*) dell'attività (i, j) è il *massimo ritardo* possibile sul suo inizio *supponendo che tutte le attività precedenti terminino al più tardi possibile*:

$$SF_{ij} = LS_{ij} - \max_{\{h\,:\,(h,\,i)\in\mathcal{A}\}}\{LF_{hi}\} = \\ = (T_j - t_{ij}) - T_i.$$

Project Scheduling: CPM time analysis

6. Percorso critico

Definizione 16: *Percorso* e *catena*

- Un *percorso* è una *catena* (sequenza) di attività che conduce dall'inizio alla fine del progetto, cioè un cammino dal nodo origine 0 al nodo finale f della rete di progetto.

Ricordiamo che (vedi Def. 3):

- Un *percorso* è *critico* se *tutte* le *attività* che lo compongono sono *critiche*.

Proprietà conseguenti

(i) In un progetto esiste *almeno un percorso critico*

(ii) Il *percorso critico* è una *catena di lunghezza massima*.

(iii) Un percorso critico è formato da soli nodi critici, ma la condizione è solo necessaria.

Project Scheduling: CPM time analysis

7. Lunghezza catena

Definizione 17: Lunghezza di una catena

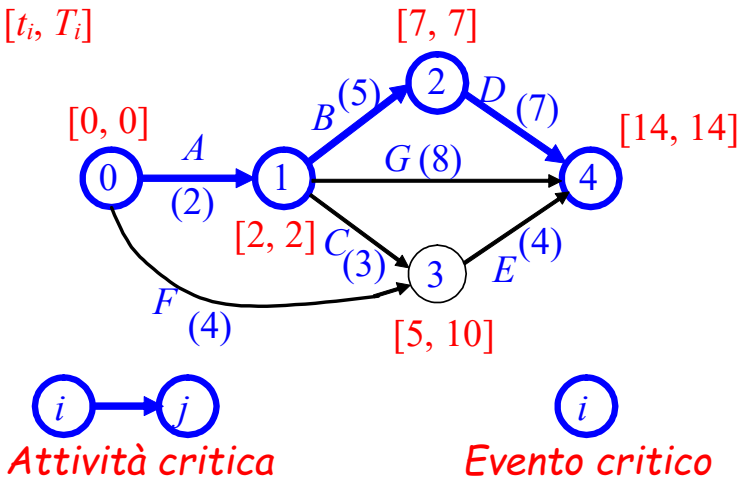
- Si definisce *lunghezza* l_α di una generica *catena* C_α dal nodo iniziale al nodo finale la grandezza

$$l_\alpha = \Sigma_{(i,j)\in C_\alpha} t_{ij}$$

- Se C_1, …, C_α, …, C_p sono tutte le catene dal nodo iniziale al nodo finale le cui corrispondenti lunghezze sono l_1, …, l_α, …, l_p, il *tempo minimo di completamento* T *del progetto* è ovviamente dato da:

$$T = \max_{1\le\alpha\le p} l_\alpha$$

Esempio



Project Scheduling: CPM time analysis

8. Percorsi sottocritici

- Nell'analisi della tempistica delle attività di un progetto può essere utile individuare i *percorsi sottocritici*
- Questi possono essere individuati nel modo seguente

  (i) Si selezionano le attività del progetto raggruppandole in funzione del loro tempo di slittamento.

  (ii) Si ordinano i gruppi di attività in ordine crescente di tempo di slittamento.

  (iii) Il primo gruppo comprende il percorso-(i) critico-(i), mentre i gruppi successivi comprendono i percorsi sottocritici in ordine crescente di distanza dalla criticità.

Osservazione

- La fase di *scheduling* di un *progetto* con *risorse illimitate* ($PS\infty|\,\text{prec}|\,C_{\max}$) corrisponde al modello di *scheduling su macchine* $P\infty|\,\text{prec}|\,C_{\max}$, in cui cioè un insieme di job soggetti a vincoli di precedenza devono essere processati su un numero illimitato di macchine parallele con l'obiettivo di minimizzare il massimo tempo di completamento dei job.

Project Scheduling: CPM time analysis

9. Formulazione come problema di PL

- Determinare il *tempo minimo di completamento T del progetto* equivale a risolvere un problema di *cammino massimo nella rete aciclica* $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ rappresentante il progetto con attività sugli archi.
- Data la rete aciclica $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ e detto t_i il tempo di raggiungimento minimo del nodo i, il problema consiste nel risolvere il seguente *problema di PL*

$$\begin{aligned} &\min\ t_f - t_0 \\ &t_j - t_i \geq t_{ij} && \forall\ (N_i, N_j) \in \mathcal{A} \\ &t_i \gtrless 0 && \forall\ N_i \in \mathcal{N} \end{aligned}$$

- Posto $\pi_i = -t_i$, riscriviamo il problema come

$$\begin{aligned} &\min\ \pi_0 - \pi_f \\ &\pi_i - \pi_j \geq t_{ij} && \forall\ (N_i, N_j) \in \mathcal{A} \\ &\pi_i \gtrless 0 && \forall\ N_i \in \mathcal{N} \end{aligned}$$

- Il problema duale è

$$\max \sum_{(N_i, N_j) \in \mathcal{A}} t_{ij}\, x_{ij}$$

$$\sum_{\{j:\,(N_i, N_j) \in \mathcal{A}\}} x_{ij} - \sum_{\{j:\,(N_j, N_i) \in \mathcal{A}\}} x_{ji} = \begin{cases} 1 & i = 0 \\ 0 & i = 1, \ldots, f-1 \\ -1 & i = f \end{cases}$$

$$x_{ij} \geq 0 \qquad \forall\ (N_i, N_j) \in \mathcal{A}$$

che è la formulazione del problema di cammino massimo (su una rete aciclica).

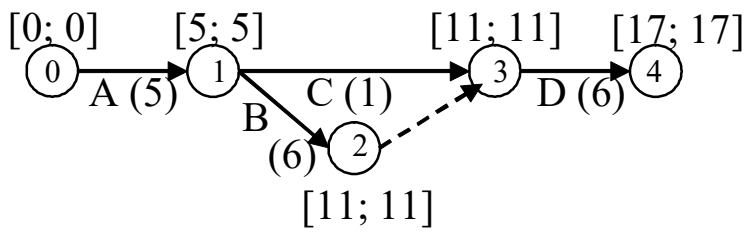
Project Scheduling: CPM time analysis

10. Dipendenza slittamenti dal grafo di progetto

Ricordiamo che:

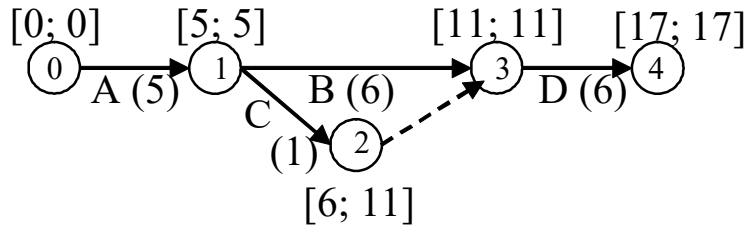
$$TF_{ij} = T_j - (t_i + t_{ij}); \quad FF_{ij} = t_j - (t_i + t_{ij}); \quad SF_{ij} = (T_j - t_{ij}) - T_i$$

$[t_i; T_j]$



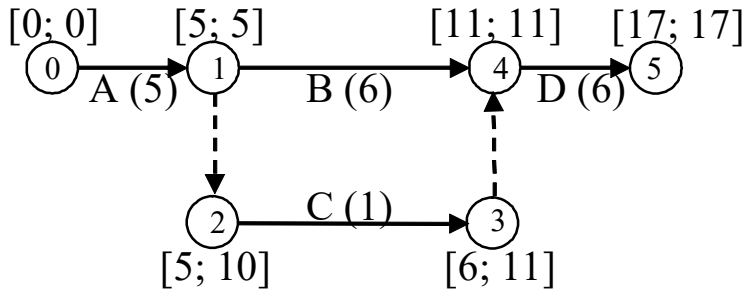
attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	5	5
D	0	0	0

$[t_i; T_j]$



attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	~~0~~	5
D	0	0	0

$[t_i; T_j]$



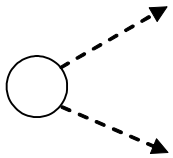
attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	~~0~~	~~0~~
D	0	0	0

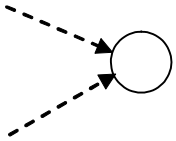
- I valori degli *slittamenti liberi FF* e di *sicurezza SF* delle attività *dipendono* dalla *rappresentazione* con grafo con attività sugli archi usato.
- Solo gli *slittamenti totali TF* sono *invarianti* rispetto alla *rappresentazione*.

Project Scheduling: CPM time analysis

10. Dipendenza slittamenti dal grafo di progetto

- I "*responsabili*" di tale difetto sono i:

nodi *out-dummy*:  *solo archi fittizi uscenti*

nodi *in-dummy*:  *solo archi fittizi entranti*

Ridefiniamo i tempi di raggiungimento minimo e massimo degli eventi, rispettivamente come:

- Sia t'_i il *minimo tempo* entro cui *almeno* un'attività che *segue* l'evento i *può* cominciare.
- Sia T'_i il *massimo tempo* entro cui *tutte* le attività che *precedono* l'evento i *devono* essere completate (pena un aumento del tempo minimo di esecuzione del progetto).

Osservazioni:

- Per i *non-dummy* $t'_i = t_i$ e $T'_i = T_i$.
- Per i *out-dummy* $t'_i \geq t_i$ e $T'_i = T_i$.
- Per i *in-dummy* $t'_i = t_i$ e $T'_i \leq T_i$.

Project Scheduling: CPM time analysis

10. Dipendenza slittamenti dal grafo di progetto

- L'algoritmo di calcolo dei tempi ai nodi è quindi:

Algoritmo di calcolo

Fase di calcolo in avanti

$t'_0 := t_0 := 0$.

for $i := 1$ to f do

$t'_i := t_i := \max_{\{k\,:\,(k,\,i)\,\in\,\mathcal{A}\}} \{t_k + t_{ki}\}$

Fase di calcolo all'indietro

$T'_f := T_f := t_f$.

for $i := f - 1$ down to 0 do

$T'_i := T_i := \min_{\{k\,:\,(i,\,k)\,\in\,\mathcal{A}\}} \{T_k - t_{ik}\}$

if i is **out-dummy** then $t'_i = \min_{\{k\,:\,(i,\,k)\,\in\,\mathcal{A}\}} \{t'_k\}$

Fase di calcolo in avanti

for $i := 1$ to f do

if i is **in-dummy** then $T'_i = \max_{\{k\,:\,(k,\,i)\,\in\,\mathcal{A}\}} \{T'_k\}$

Complessità delle tre fasi $O(m)$, dove m è il numero di archi della rete di progetto.

Osservazioni:

- Per le attività (i, j) *non-fittizie* $t'_i = t_i$, $T'_j = T_j$ e:

$$ES_{ij} = t'_i \;;\; LS_{ij} = T'_j - t_{ij} \;;\quad EF_{ij} = t'_i + t_{ij} \;;\; LF_{ij} = T'_j$$

- Gli slittamenti totali $TF_{ij} = T'_j - t'_i - t_{ij}$ delle attività (i, j) *non-fittizie* sono comunque invarianti rispetto alla rappresentazione.
- Gli slittamenti ai nodi i vanno calcolati con i tempi originari: $F_i = T_i - t_i$

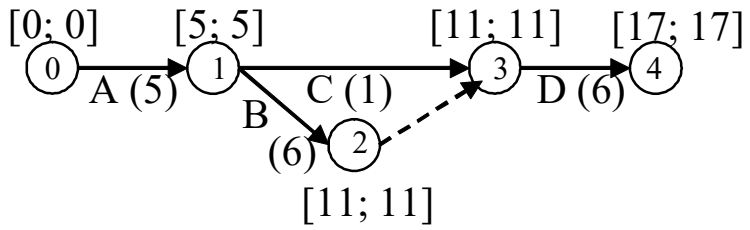
Project Scheduling: CPM time analysis

10. Dipendenza slittamenti dal grafo di progetto

Ricalcolando tutti gli slittamenti secondo i nuovi tempi sui nodi:

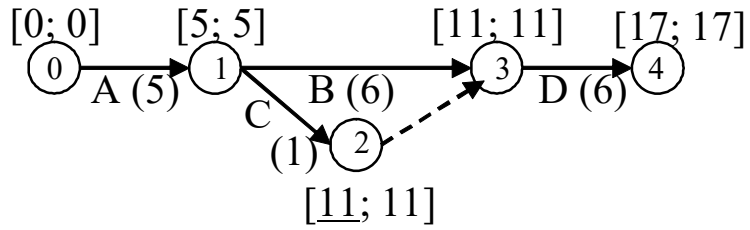
$$TF_{ij} = T'_j - (t'_i + t_{ij}); \quad FF_{ij} = t'_j - (t'_i + t_{ij}); \quad SF_{ij} = (T'_j - t_{ij}) - T'_i$$

$[t'_i; T'_j]$



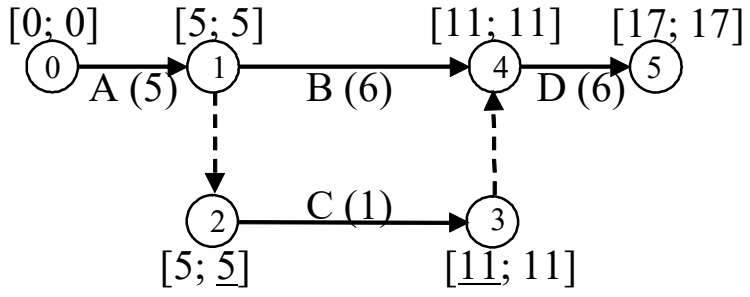
attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	5	5
D	0	0	0

$[t'_i; T'_j]$



attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	5	5
D	0	0	0

$[t'_i; T'_j]$



attività	*TF*	*FF*	*SF*
A	0	0	0
B	0	0	0
C	5	5	5
D	0	0	0

- Come si può notare, *in tal modo anche* gli *slittamenti liberi FF* e di *sicurezza SF* delle attività sono *invarianti* rispetto alla *rappresentazione* con grafo con attività sugli archi usato.

Project Scheduling: CPM time analysis

11. Reti AON: Calcolo tempi minimi e massimi d'inizio e completamento delle attività

- Siano $2, \ldots, n-1$, le $n-2$ attività (vere) del progetto.
- 1 e n attività *dummy* aggiuntive di durata 0: la 1 è l'unica senza predecessori e la n è l'unica senza successori.
  - Indichiamo con:
    - p_i: *durata* dell'attività i;
    - P_i: insieme *predecessori diretti* dell'attività i;
    - S_i: insieme *successori diretti* dell'attività i.
  - I valori di:
    - ES_i: (*earliest start time*), min. istante d'inizio di i
    - LF_i: (*latest finish time*), max. istante di fine di i
    - EF_i: (*earliest finish time*), min. istante di fine di i
    - LS_i: (*latest start time*), max. istante d'inizio di i
  - Si calcolano con procedure *forward* e *backward*
- *procedura forward*

  $ES_1 := EF_1 := 0;$

  for $i := 2$ to n do

  $ES_i := \max [EF_j \mid j \in P_i],$

  $EF_i := ES_i + p_i.$

  La durata minima del progetto è $T = EF_n = ES_n$.
- *procedura backward*

  $LF_n := LS_n := T;$

  for $i := n-1$ down to 1 do

  $LF_i := \min [LS_j \mid j \in S_i],$

  $LS_i := LF_i - p_i.$