2JSP: Condivisione a tempo minimo di risorse per due lavori*. Griglia, Grafo degli stati

*Per la quasi totalità di questa unità didattica si possono prendere come riferimento le dispense di A. Agnetis: ”Dispense di Automazione Industriale per il corso Universitario”, ad uso interno.1993

2JSP: two job shop problem

Cioè problema di due lavori da svolgere in un'officina

I due lavori sono costituiti ciascuno da una sequenza di operazioni non interrompibili.

Alcune o tutte le operazioni richiedono una risorsa (macchina, utensile, stazione di lavoro, ma anche sw, operatore umano, accesso dati, …..) in uso esclusivo, con tempo di esecuzione assegnato.

Alcune risorse sono da condividere, in divisione di tempo, tra due o più operazioni, con la possibilità di creare conflitti, se appartengono a lavori diversi.

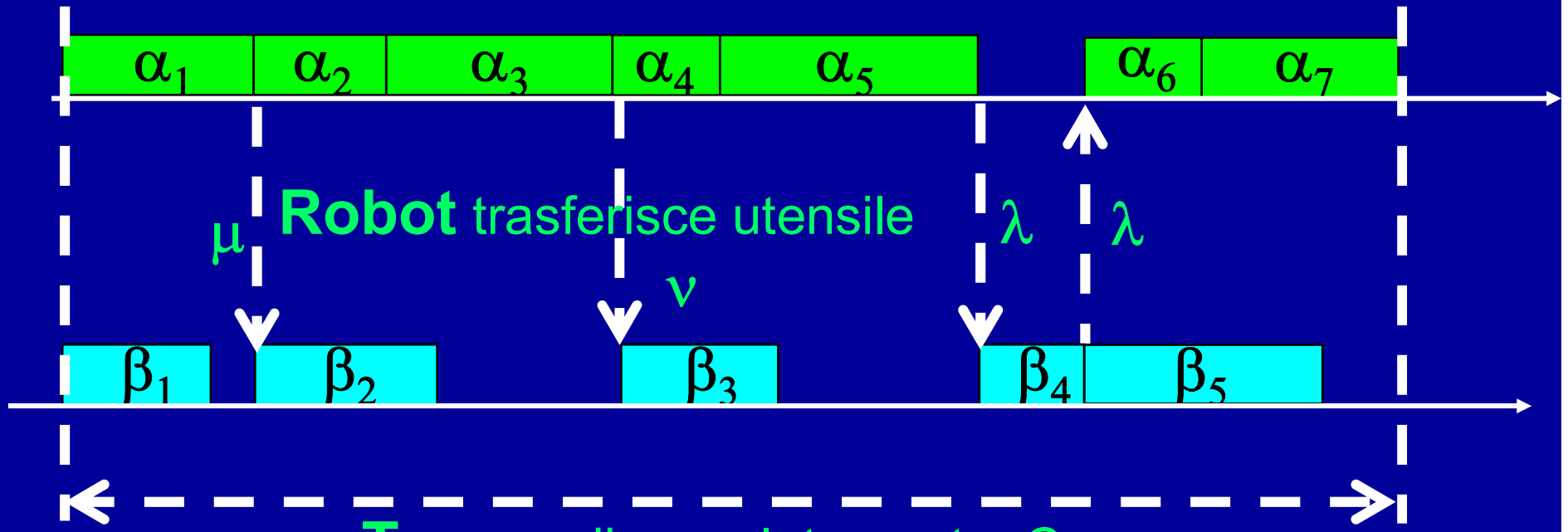
Il problema è minimizzare il tempo complessivo di completamento di entrambi i lavori, commutando cioè in modo ottimo (in tempo trascurabile o meno) l'impiego della risorsa da condividere da un lavoro all'altro, in caso di conflitto.

ESEMPIO 2JSP1: Due lavori: A e B

t.A	ut.A	t.B	ut.B	indici
α	cond.	β	cond.	op.
2,5	μ	2	ρ	1
1,7		2,4	μ	2
3,1	ν	2,1	ν	3
1,6	ρ	1,4	λ	4
3,5	λ	3,1	ν	5
1,4	λ			6
2,5	λ			7

CELLA CON UTENSILI CONDIVISI:
condivisione a tempo minimo: Gantt
Tempo di trasferimento trascurabile

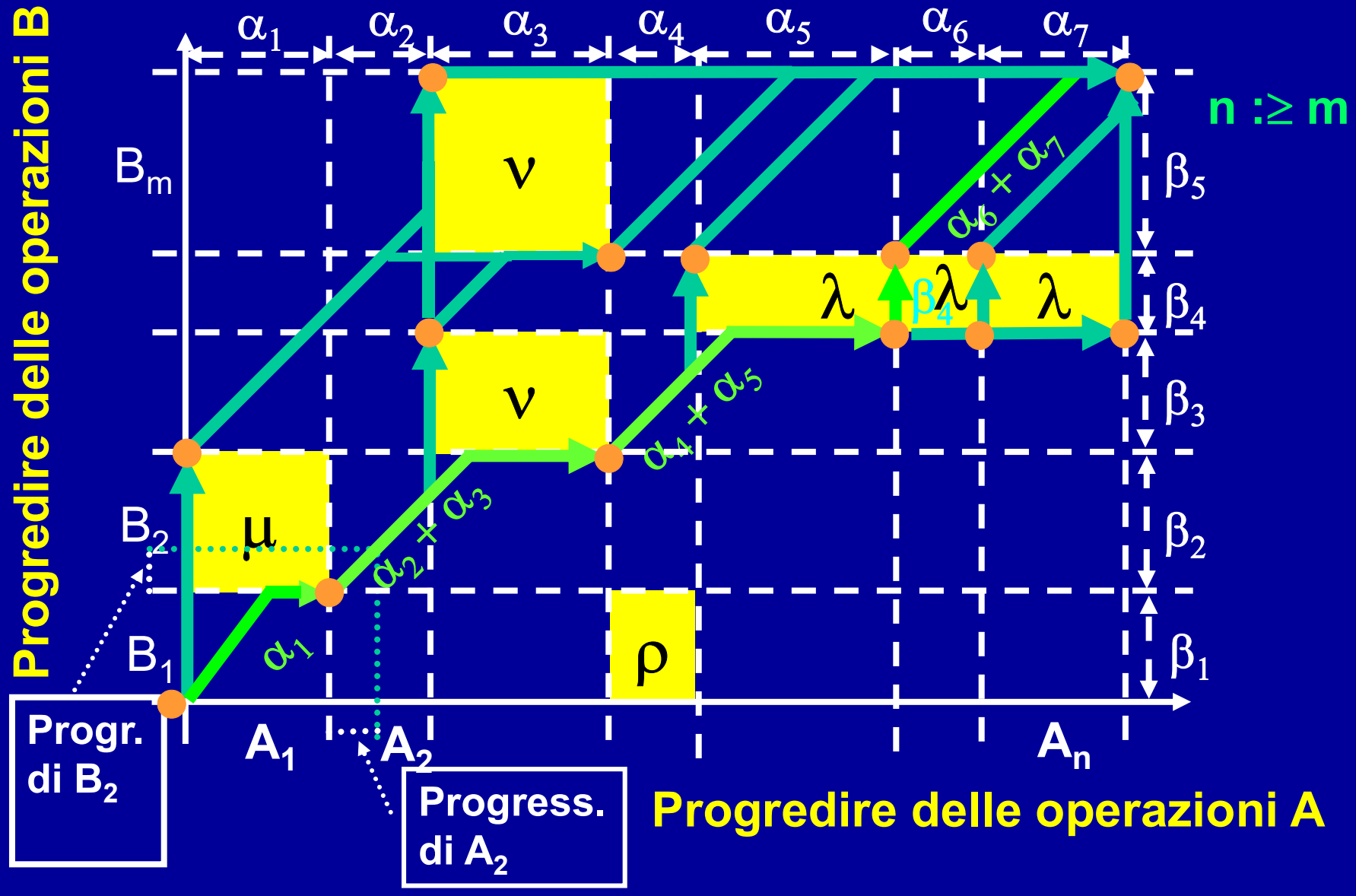
(Si veda anche Agnetis "Dispense …": 2.3 scheduling .. pp. 49_ _53)



Sequenziamento greedy (famelico): le operazioni iniziano appena possibile, cioè prende prima l'utensile chi lo chiede prima (minimizza Cm?)

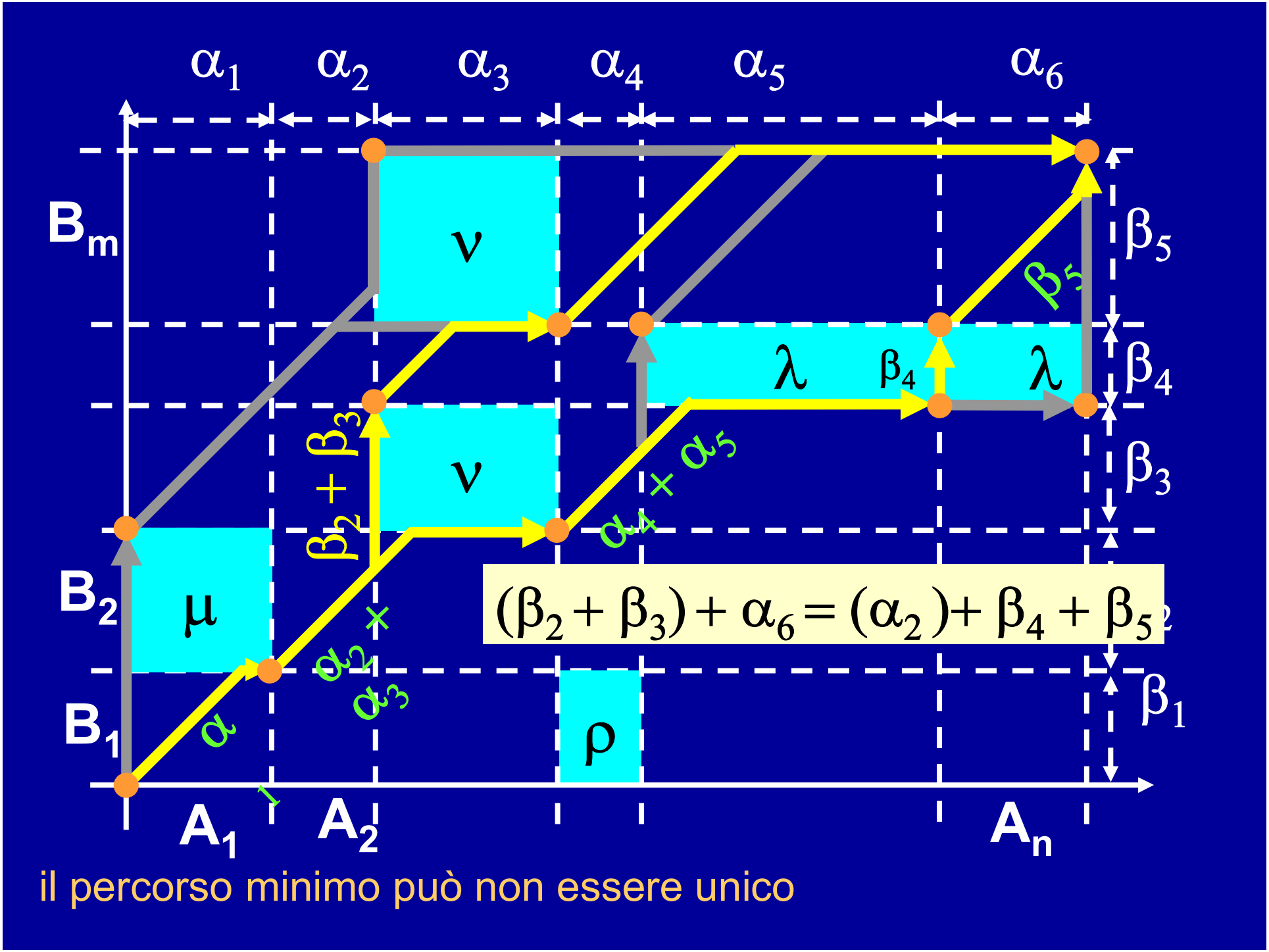
GRIGLIA DELLE OPERAZIONI

Tempo robot trascurabile: Condivisione a tempo minimo



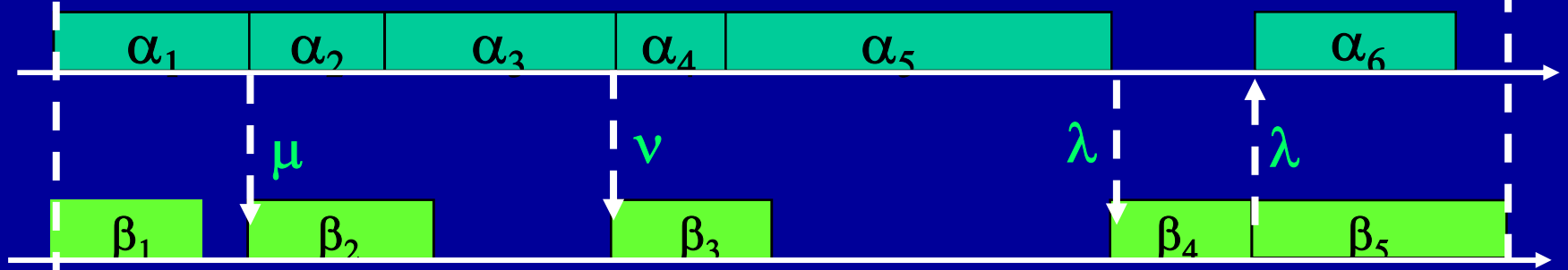
ESEMPIO 2JSP2: Due lavori assegnati: A e B

t.A	ut.A	t.B	ut.B	indici
α	cond.	β	cond.	op.
14	μ	12	ρ	1
10		16	μ	2
16	ν	14	ν	3
18	ρ	9	λ	4
31	λ	20	ν	5
14	λ			6

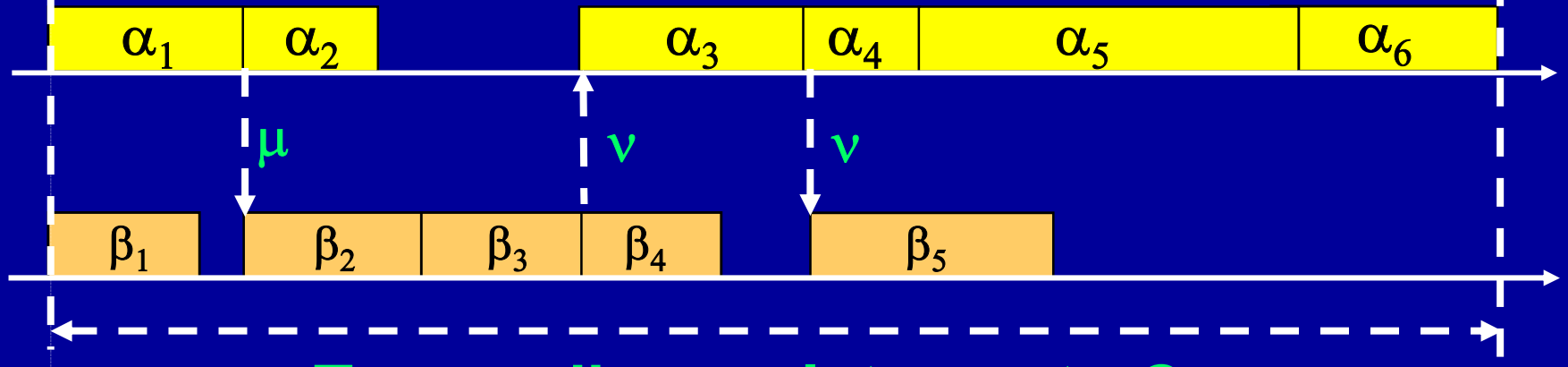


il percorso minimo può non essere unico

CELLA CON UTENSILI CONDIVISI



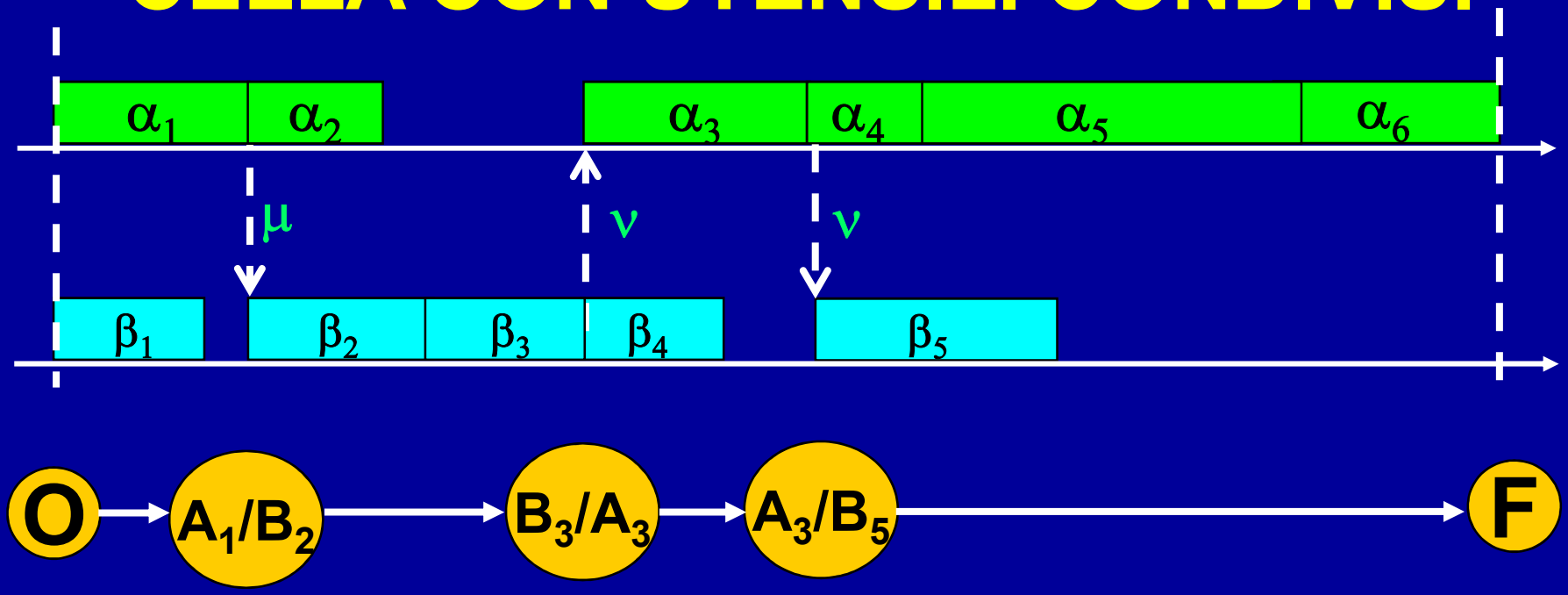
Schedule equivalente:



Tempo di completamento Cm

il percorso minimo può non essere unico: qui due seq. ottimi

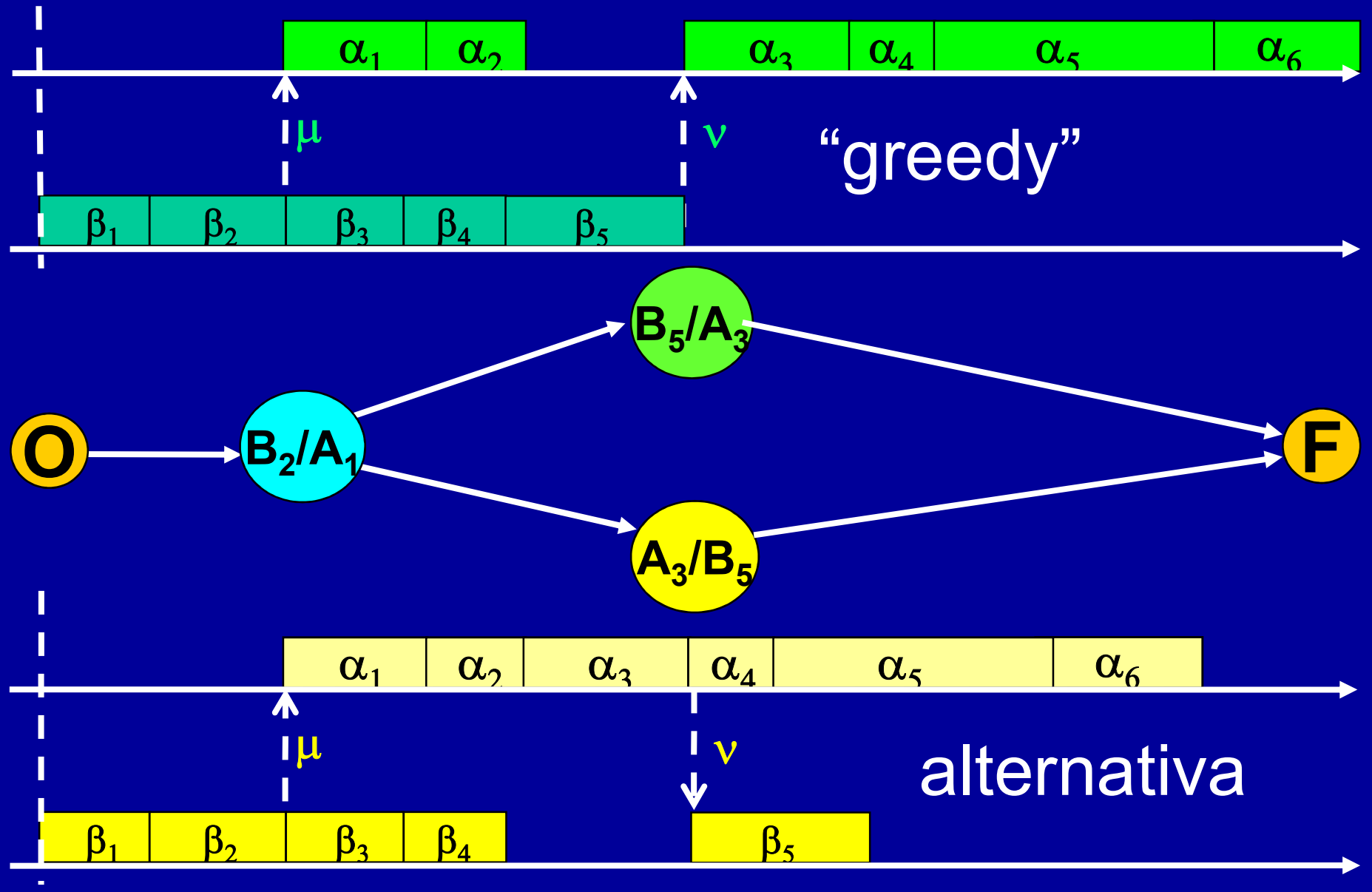
CELLA CON UTENSILI CONDIVISI

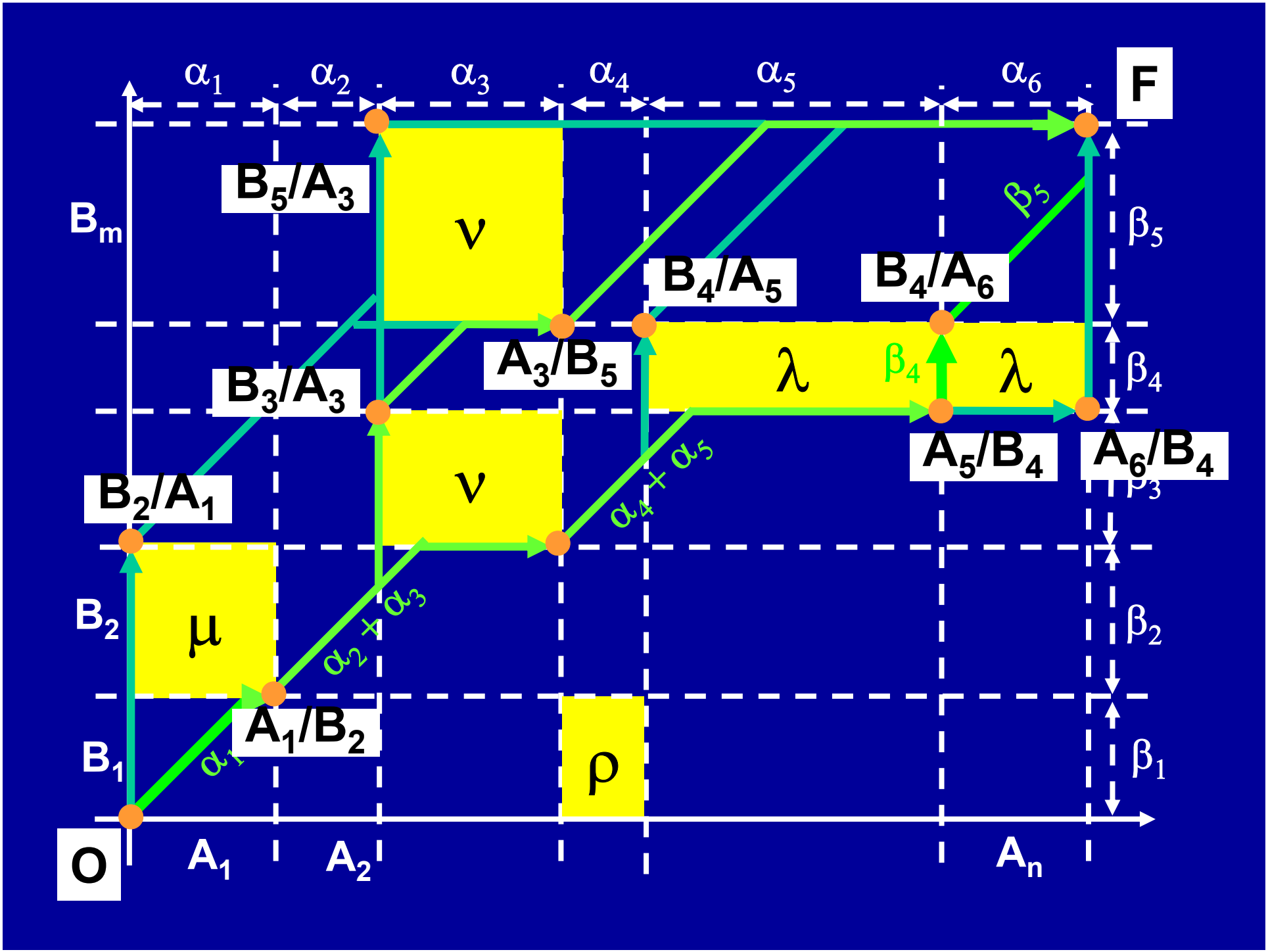


GLI SCAMBI DI UTENSILE CORRISPONDONO A PARTICOLARI STATI DEL SISTEMA
(stati di commutazione)

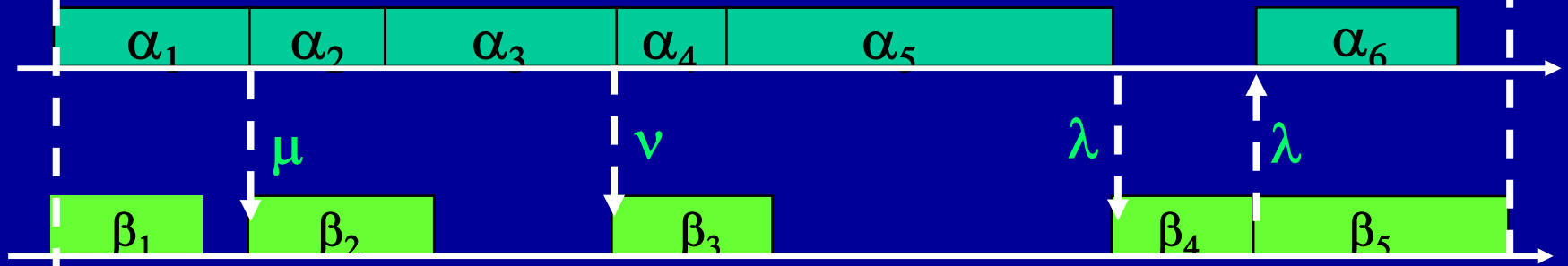
Il grafo di stato si può ottenere direttamente dal diagramma di Gantt

SCAMBIO UTENSILI : STATI DAL GANTT

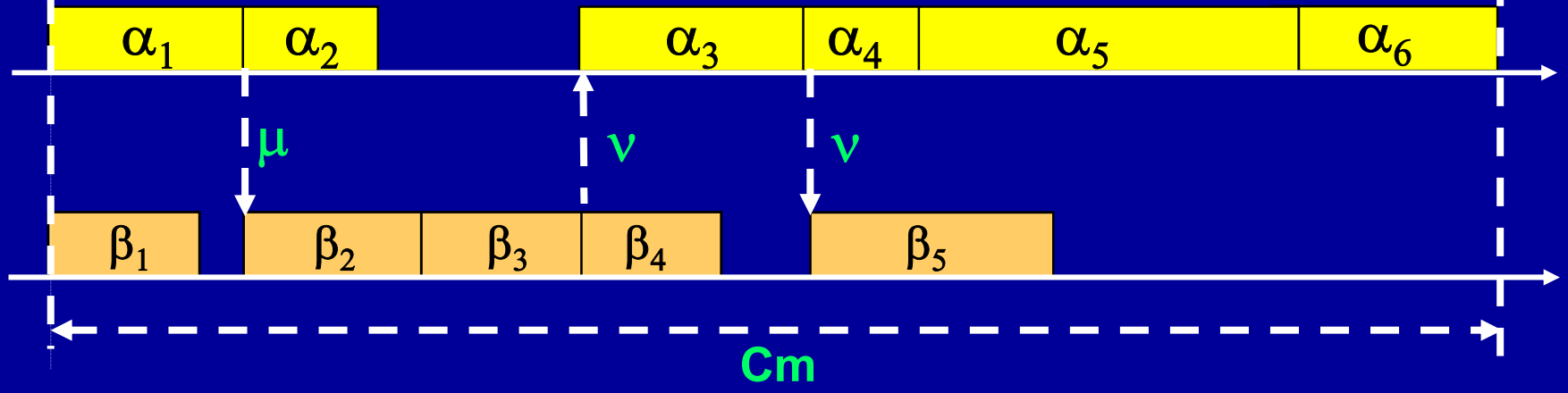



α1
α2
α3
α4
α5
α6
F
Bm
B5/A3
ν
β5
β5
B4/A5
B4/A6
A3/B5
λ
β4
λ
β4
B3/A3
ν
A5/B4
A6/B4
β3
B2/A1
α4 + α5
B2
μ
α2 + α3
β2
A1/B2
B1
α1
ρ
β1
O
A1
A2
An

CELLA CON UTENSILI CONDIVISI

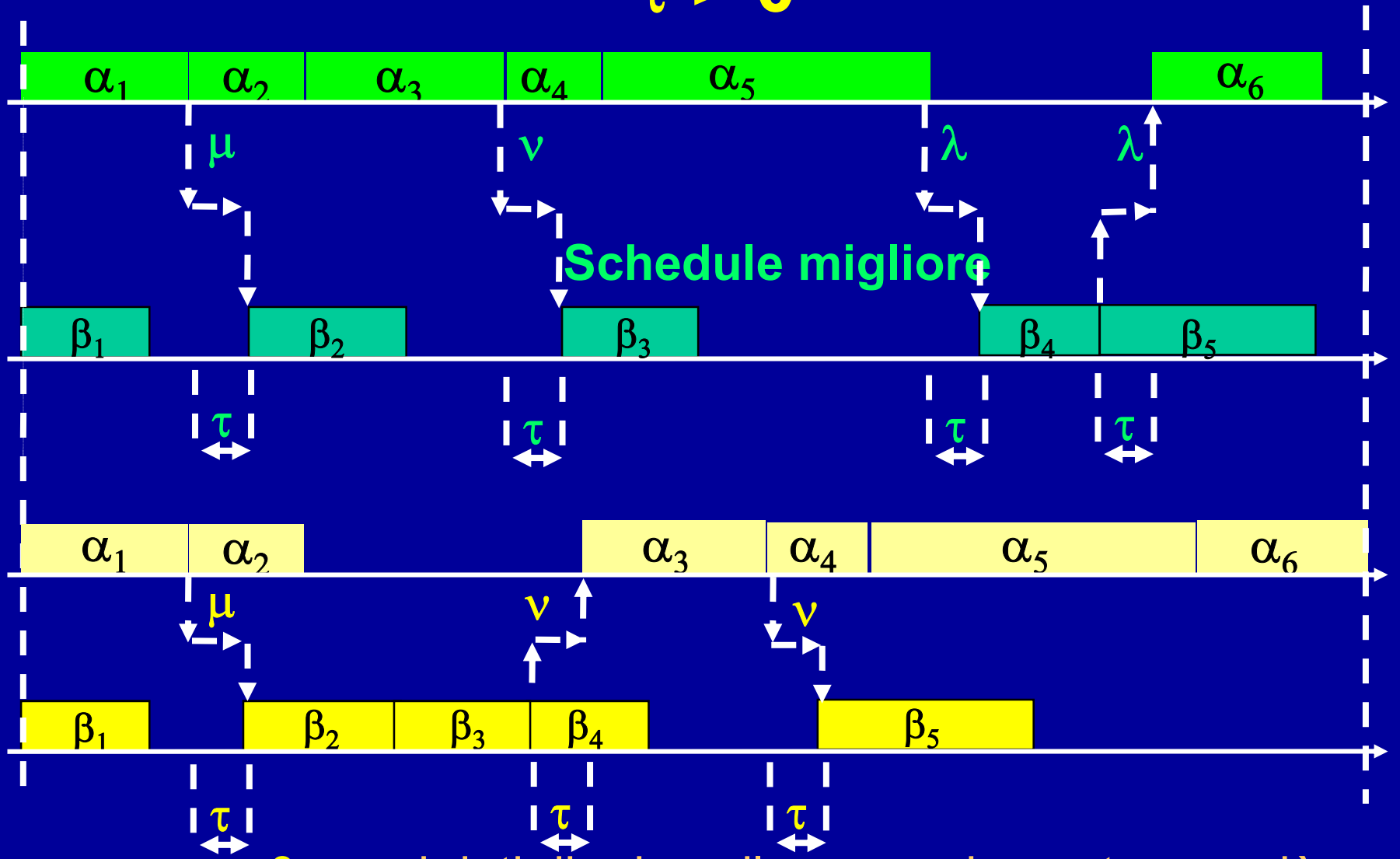


Schedule equivalente:



Si supponga non trascurabile il tempo di commutazione

UTENSILI CONDIVISI: tempo robot $\tau > 0$



se $\tau > 0$, con i dati di prima, il sequenziamento con più commutazioni è il migliore

ALGORITMO A*

La procedura di costruzione del cammino minimo e' simile a quella dell'algoritmo di Dijkstra. Utilizza però, per l'espansione di un nodo, una stima h* del costo di un cammino che è tra i migliori tra quelli che passano per il nodo dato

f (i):= costo di un miglior cammino da i ad F

f* (i):= stima del costo di un miglior cammino da i ad F

d*(i):= costo di un miglior cammino da O a i finora trovato

$$h^*(i) := d^*(i) + f^*(i)$$

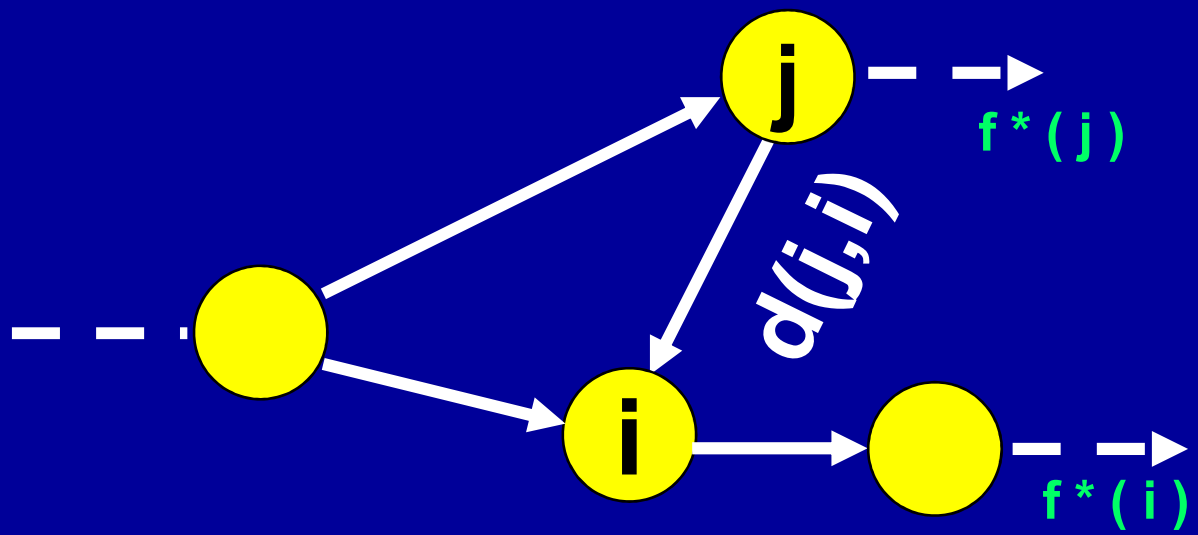
$$h^*(i) = d^*(i) + f^*(i)$$

$h^*(i)$ stima del costo di un miglior cammino, passante per i, che unisce O ad F e che segue fino a i il cammino di costo minimo individuato

Se $f^* \leq f \Rightarrow$ A* è ammissibile

Cioè

Se f * è una stima per difetto di f allora A* trova un cammino ottimo verso F, se ne esiste uno



ASSUNZIONE DI CONSISTENZA: passando per un successore non peggiora la stima, cioè:

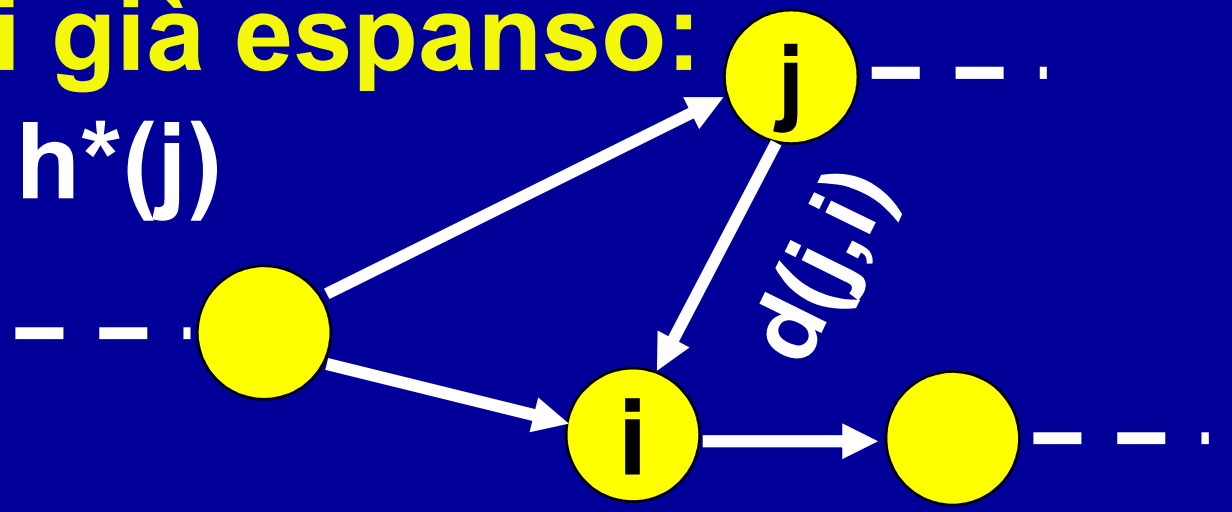
$$f^*(j) \leq d(j,i) + f^*(i)$$

Allora, applicato come Dijkstra, A* dà la distanza minima dall'origine, perché risulta:

$$h^*(i) \leq h^*(j) \Rightarrow d^*(i) \leq d^*(j) + d(j,i)$$

Nodo i già espanso:

$h^*(i) \leq h^*(j)$



CONSISTENZA:

$$d^*(j) + f^*(j) \leq d^*(j) + f^*(i) + d(j,i)$$

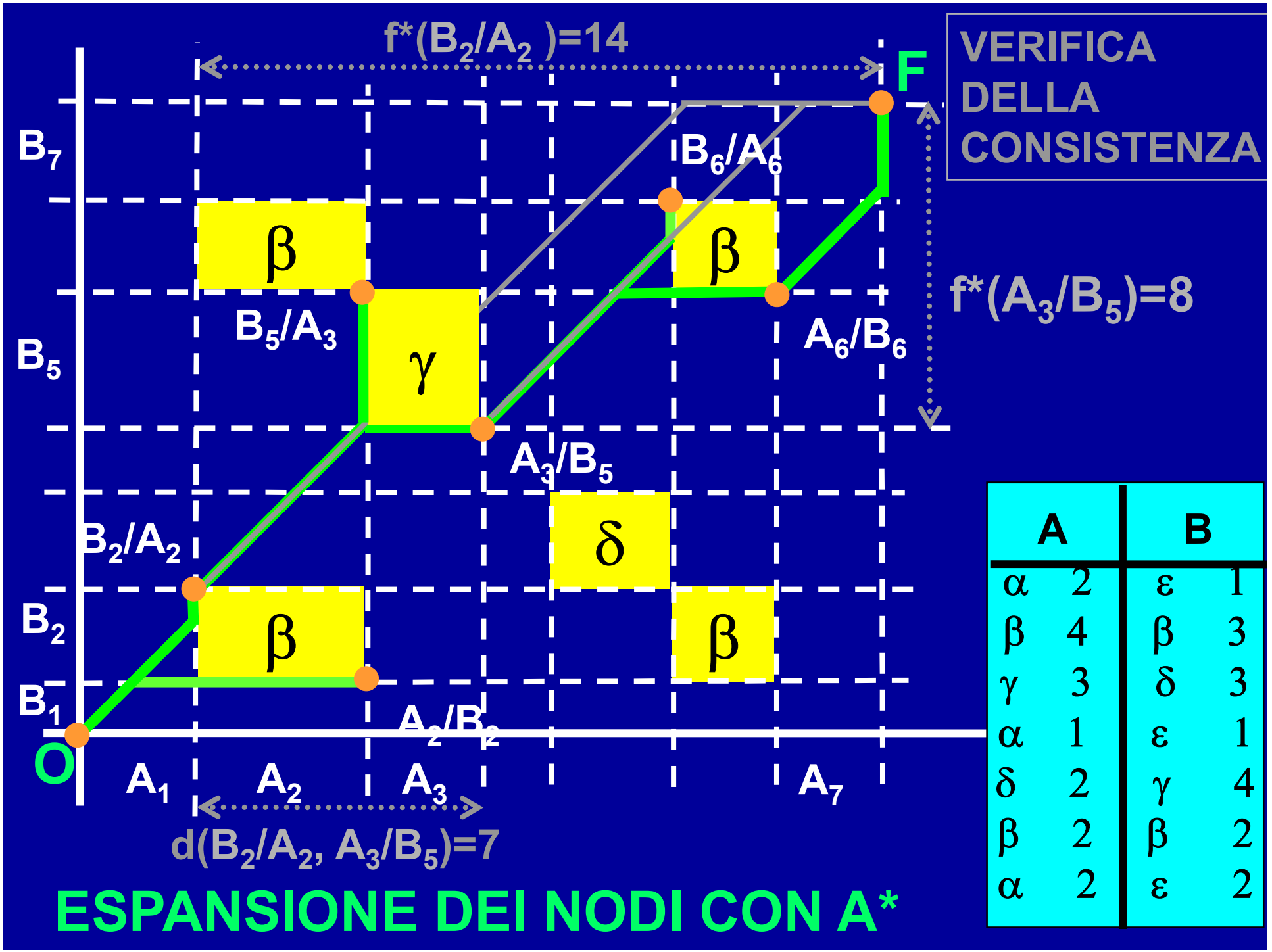
da cui, se $h^*(i) \leq h^*(j)$ **cioè se:**

$d^*(i) + f^*(i) \leq d^*(j) + f^*(j)$ **si ha:**

$$d^*(i) + f^*(i) \leq d^*(j) + f^*(i) + d(j,i)$$

$$h^*(i) \leq h^*(j) \Rightarrow d^*(i) \leq d^*(j) + d(j,i)$$

Cioè non posso raggiungere un nodo già espanso con distanza minore, quindi la sua distanza minima dall'origine è d(i)=d*(i).



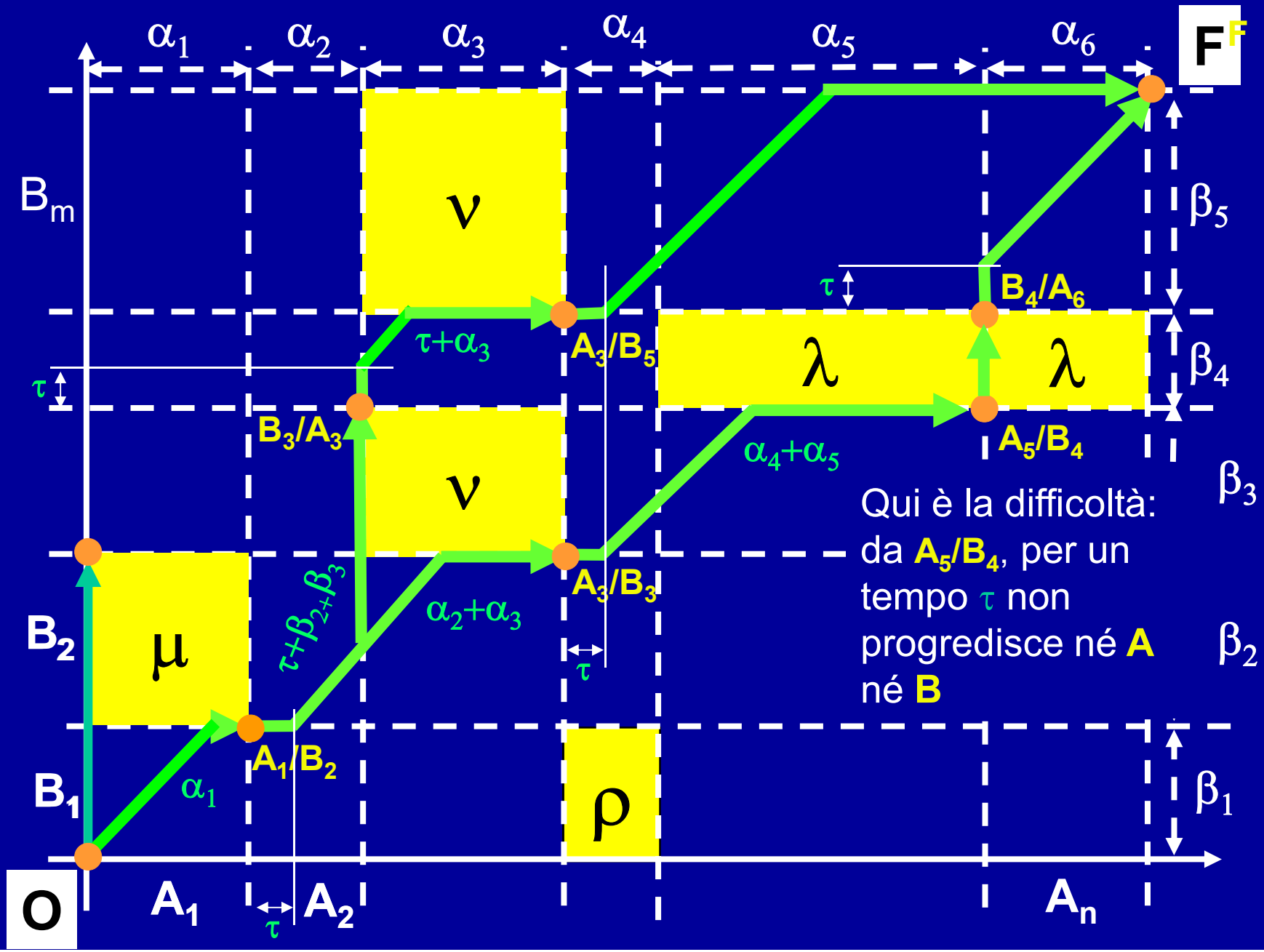
A		B	
α	2	ε	1
β	4	β	3
γ	3	δ	3
α	1	ε	1
δ	2	γ	4
β	2	β	2
α	2	ε	2

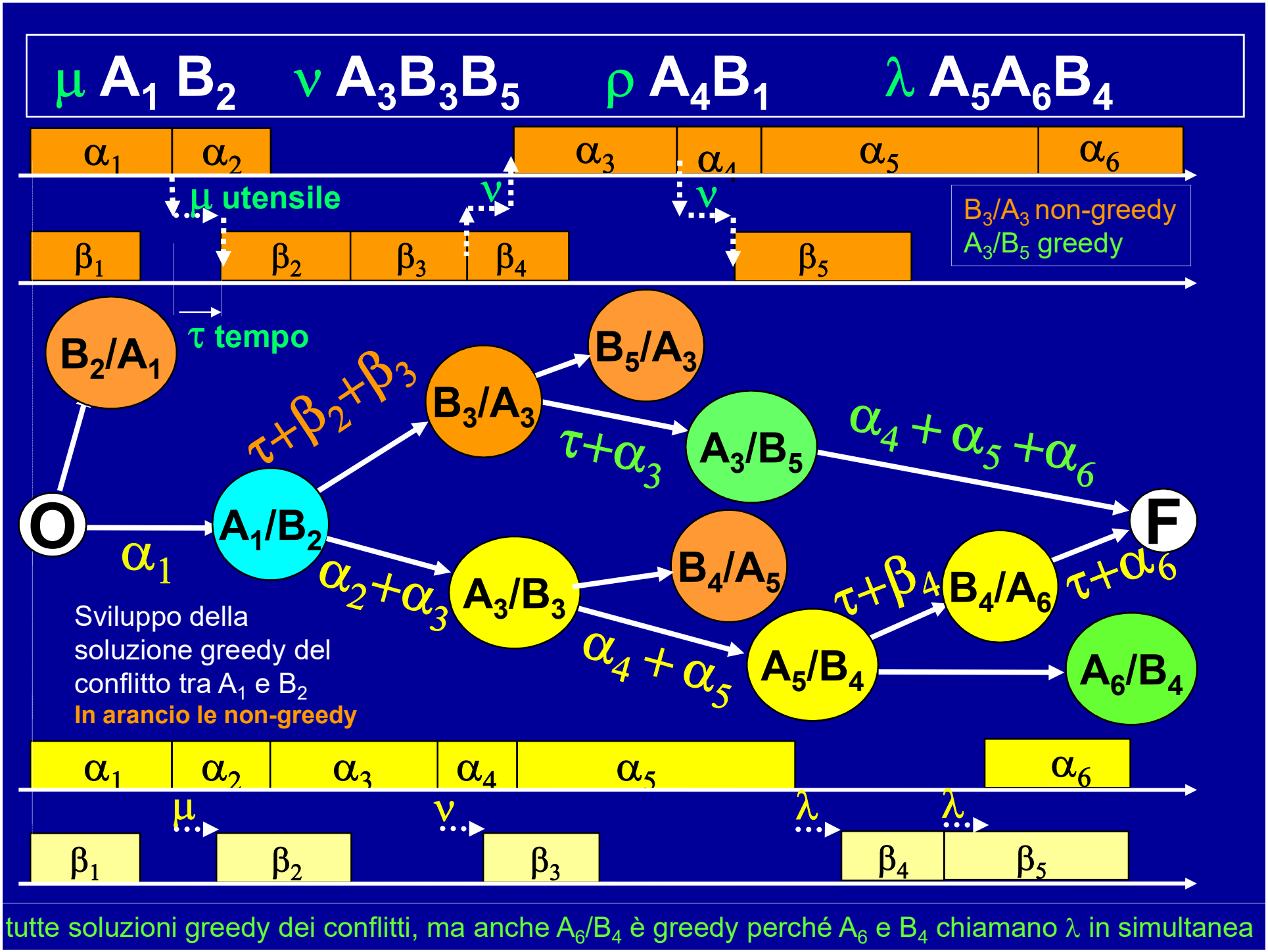
UTENSILI CONDIVISI:
tempo robot $\tau > 0$

Non si può usare la griglia,
ma si può ugualmente costruire un
GRAFO DI STATO

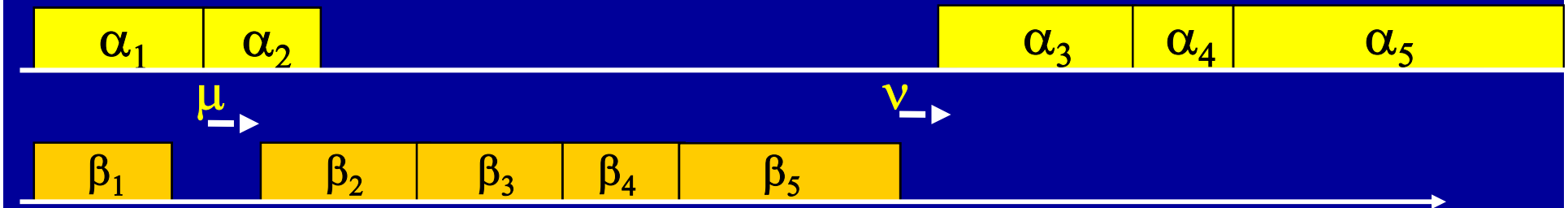
Si sviluppa prima il sequenziamento
con soluzione "greedy" dei conflitti,
quindi si sviluppano i sequenziamenti
con soluzioni alternative

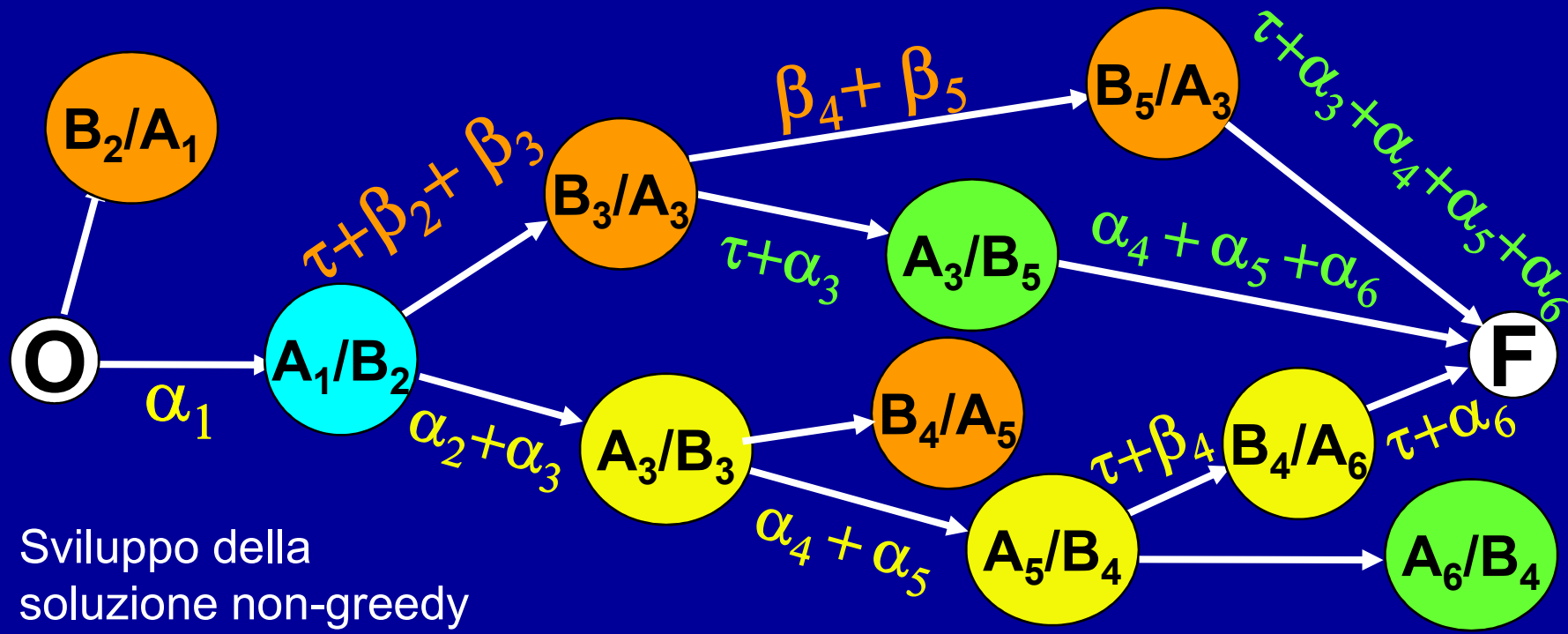
Difficoltà di rappresentare sulla griglia il tempo τ



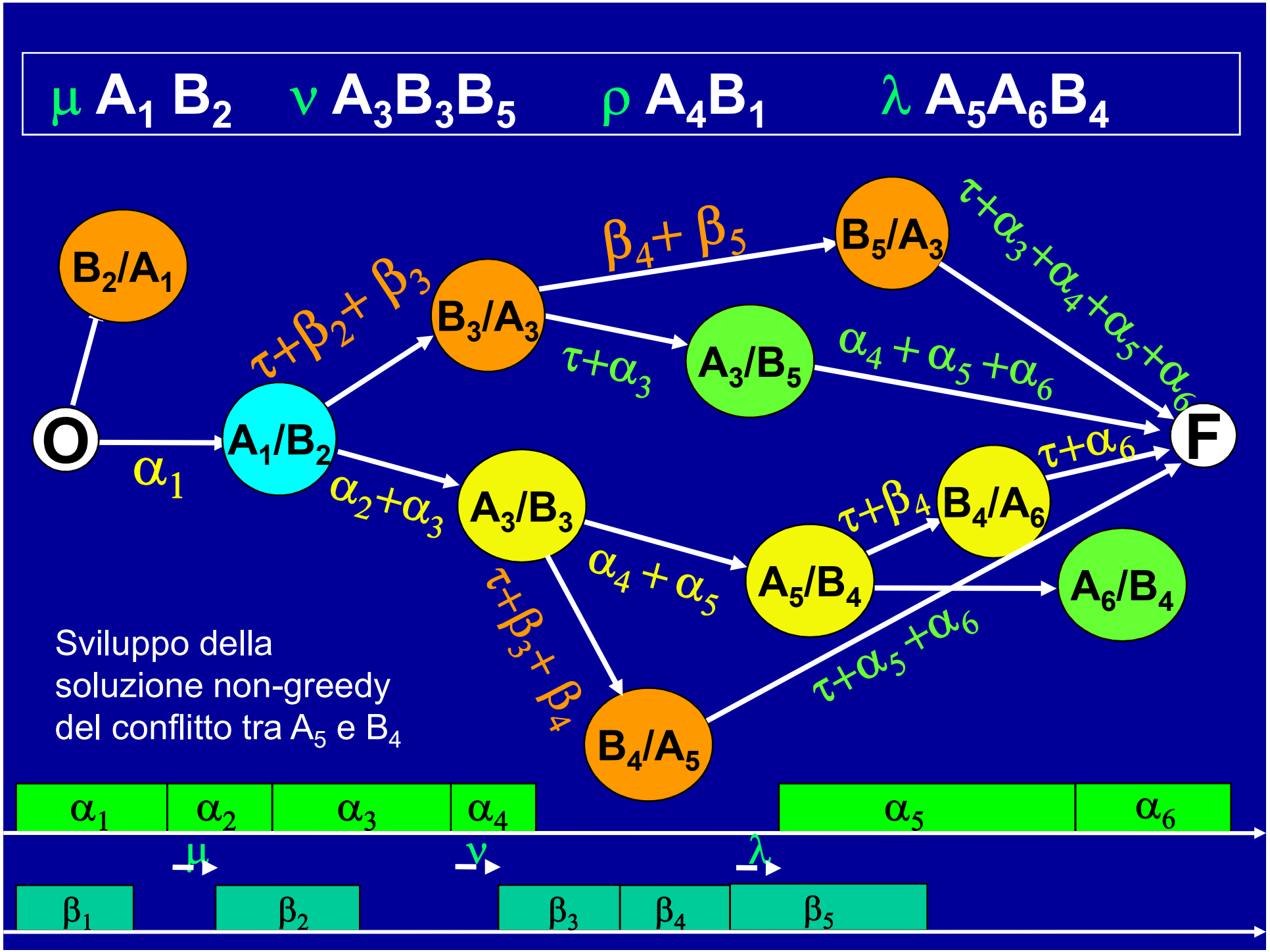

μ A1 B2
ν A3B3B5
ρ A4B1
λ A5A6B4
α1
α2
α3
α4
α5
α6
μ utensile
ν
ν
β1
β2
β3
β4
β5
B3/A3 non-greedy
A3/B5 greedy
τ tempo
B2/A1
B3/A3
B5/A3
τ+β2+β3
τ+α3
A3/B5
α4 + α5 +α6
O
A1/B2
α1
α2+α3
A3/B3
B4/A5
α4 + α5
A5/B4
τ+β4
B4/A6
τ+α6
F
A6/B4
Sviluppo della soluzione greedy del conflitto tra A1 e B2
In arancio le non-greedy
α1
α2
α3
α4
α5
α6
μ
ν
λ
λ
β1
β2
β3
β4
β5
tutte soluzioni greedy dei conflitti, ma anche A6/B4 è greedy perché A6 e B4 chiamano λ in simultanea

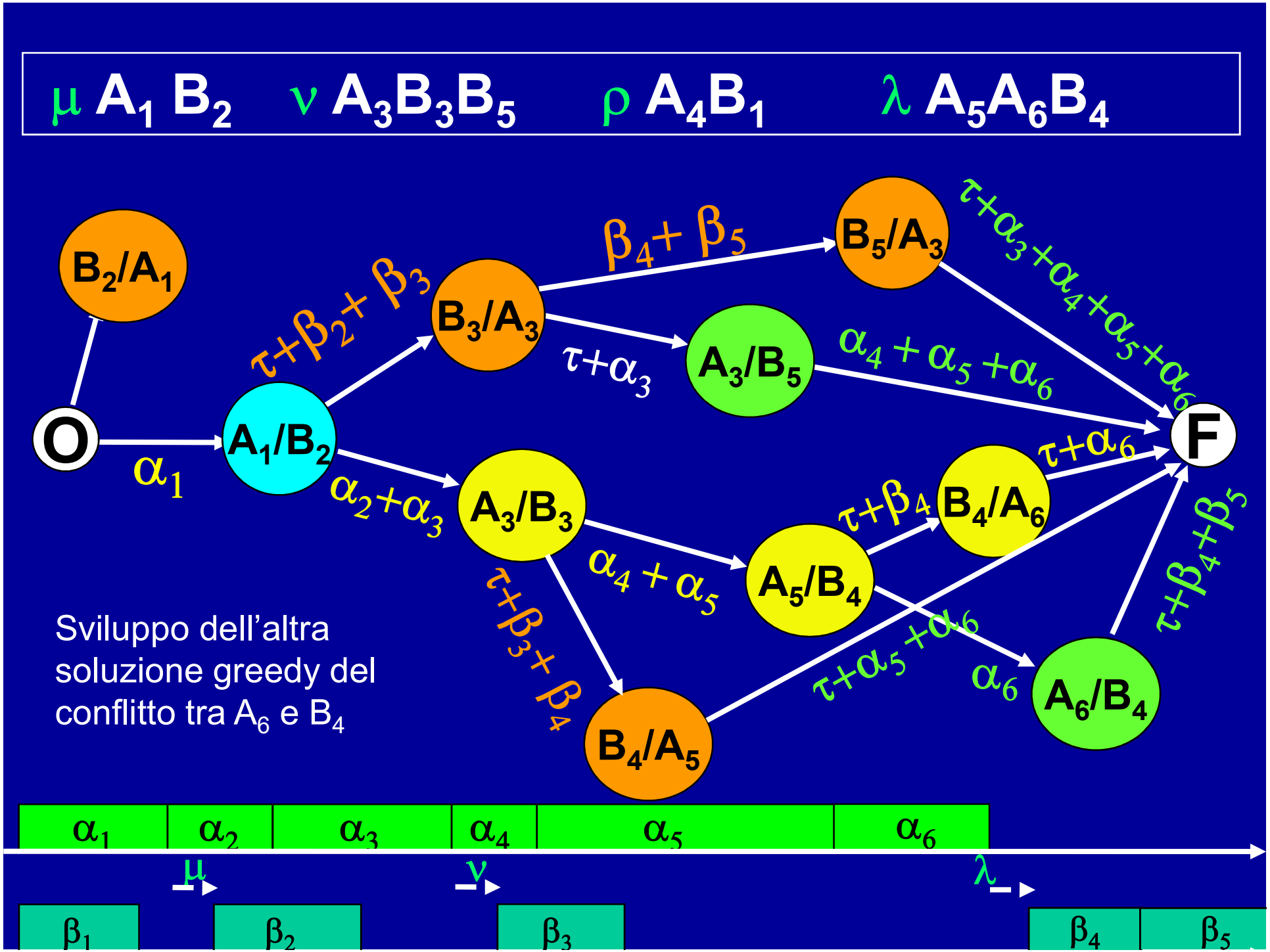
μ A_1 B_2 ν $A_3B_3B_5$ ρ A_4B_1 λ $A_5A_6B_4$

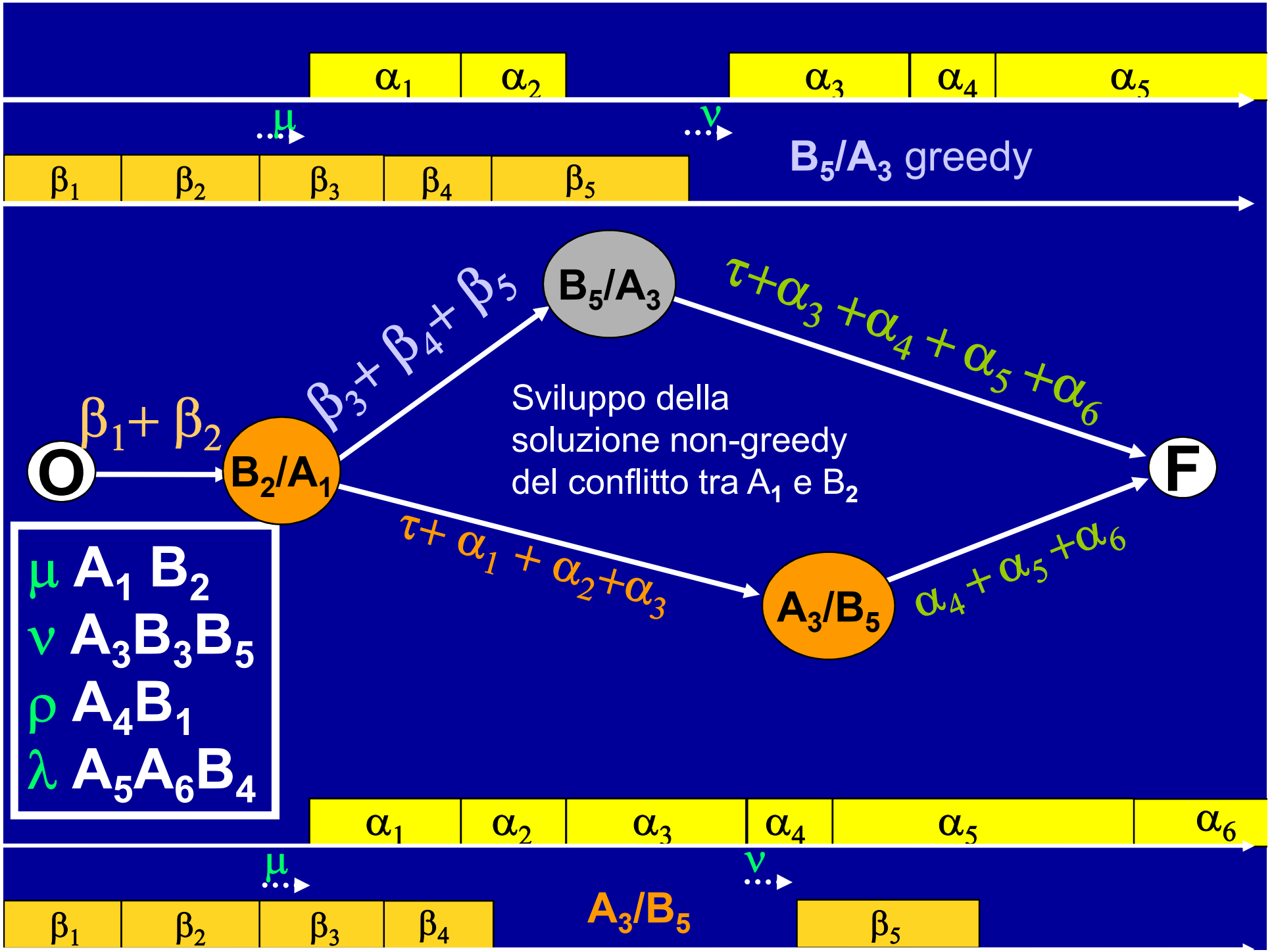




Sviluppo della soluzione non-greedy del conflitto tra A_3 e B_5


μ A1 B2
ν A3B3B5
ρ A4B1
λ A5A6B4
B2/A1
O
α1
A1/B2
τ+β2+ β3
B3/A3
β4+ β5
B5/A3
τ+α3+α4+α5+α6
τ+α3
A3/B5
α4 + α5 +α6
F
α2+α3
A3/B3
α4 + α5
τ+β3+ β4
B4/A5
A5/B4
τ+β4
B4/A6
τ+α6
A6/B4
τ+α5 +α6
Sviluppo della
soluzione non-greedy
del conflitto tra A5 e B4
α1
α2
α3
α4
α5
α6
μ
ν
λ
β1
β2
β3
β4
β5


μ A1 B2
ν A3B3B5
ρ A4B1
λ A5A6B4
O
F
B2/A1
A1/B2
B3/A3
A3/B5
B5/A3
A3/B3
A5/B4
B4/A6
A6/B4
B4/A5
α1
τ+β2+ β3
β4+ β5
τ+α3
α4 + α5 +α6
τ+α3+α4+α5+α6
α2+α3
α4 + α5
τ+β4
τ+α6
τ+β3+ β4
τ+α5 +α6
α6
τ+β4+β5
Sviluppo dell'altra soluzione greedy del conflitto tra A6 e B4
α1
α2
α3
α4
α5
α6
μ
ν
λ
β1
β2
β3
β4
β5


α1
α2
α3
α4
α5
μ
ν
B5/A3 greedy
β1
β2
β3
β4
β5
B5/A3
β3+ β4+ β5
τ+α3 +α4 + α5 +α6
Sviluppo della
soluzione non-greedy
del conflitto tra A1 e B2
β1+ β2
O
B2/A1
F
τ+ α1 + α2+α3
A3/B5
α4 + α5 +α6
μ A1 B2
ν A3B3B5
ρ A4B1
λ A5A6B4
α1
α2
α3
α4
α5
α6
μ
ν
β1
β2
β3
β4
A3/B5
β5

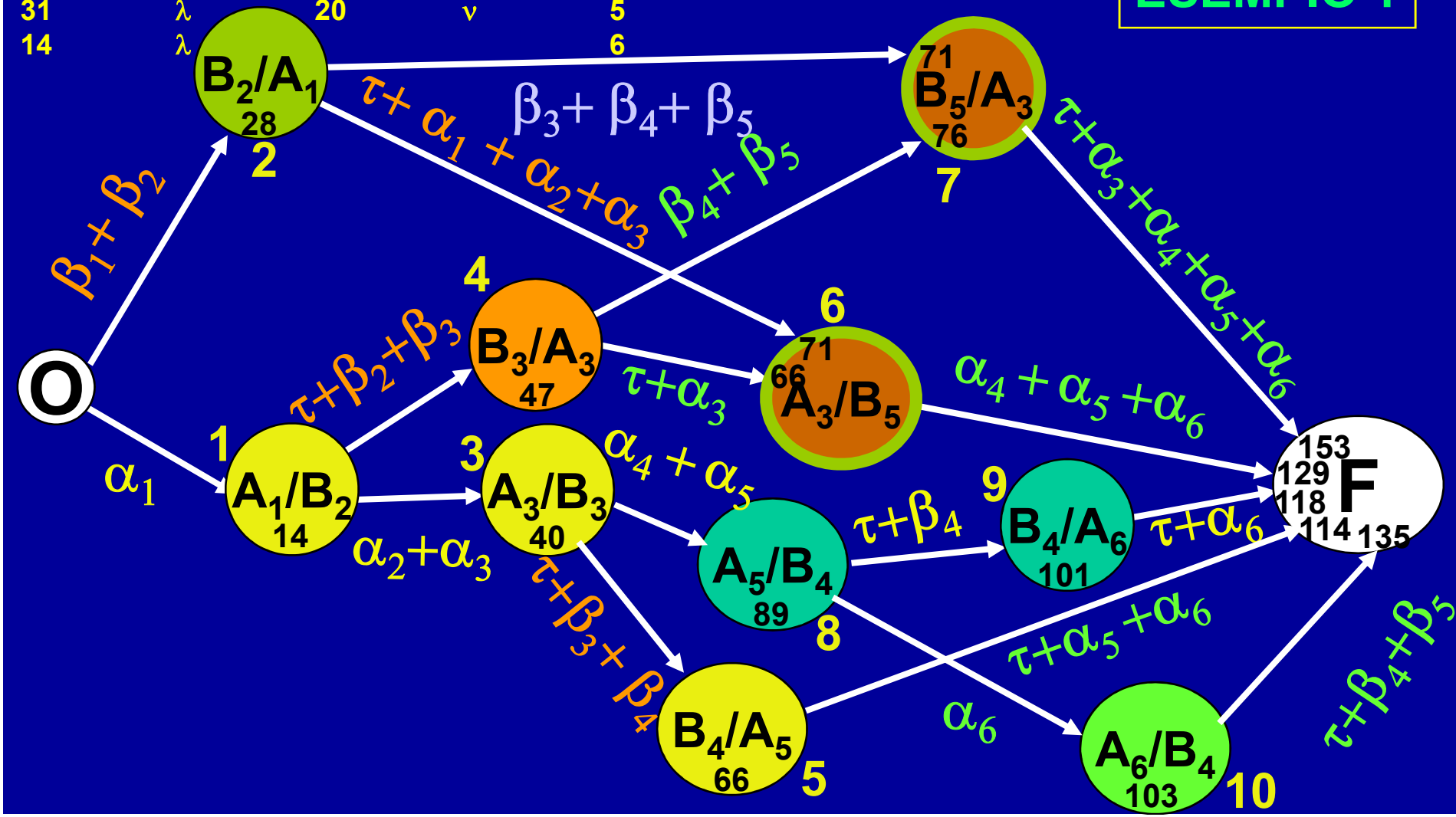
ESEMPIO 2JSP1: Due lavori assegnati: A e B
Tempo di commutazione degli utensili $\tau = 3$

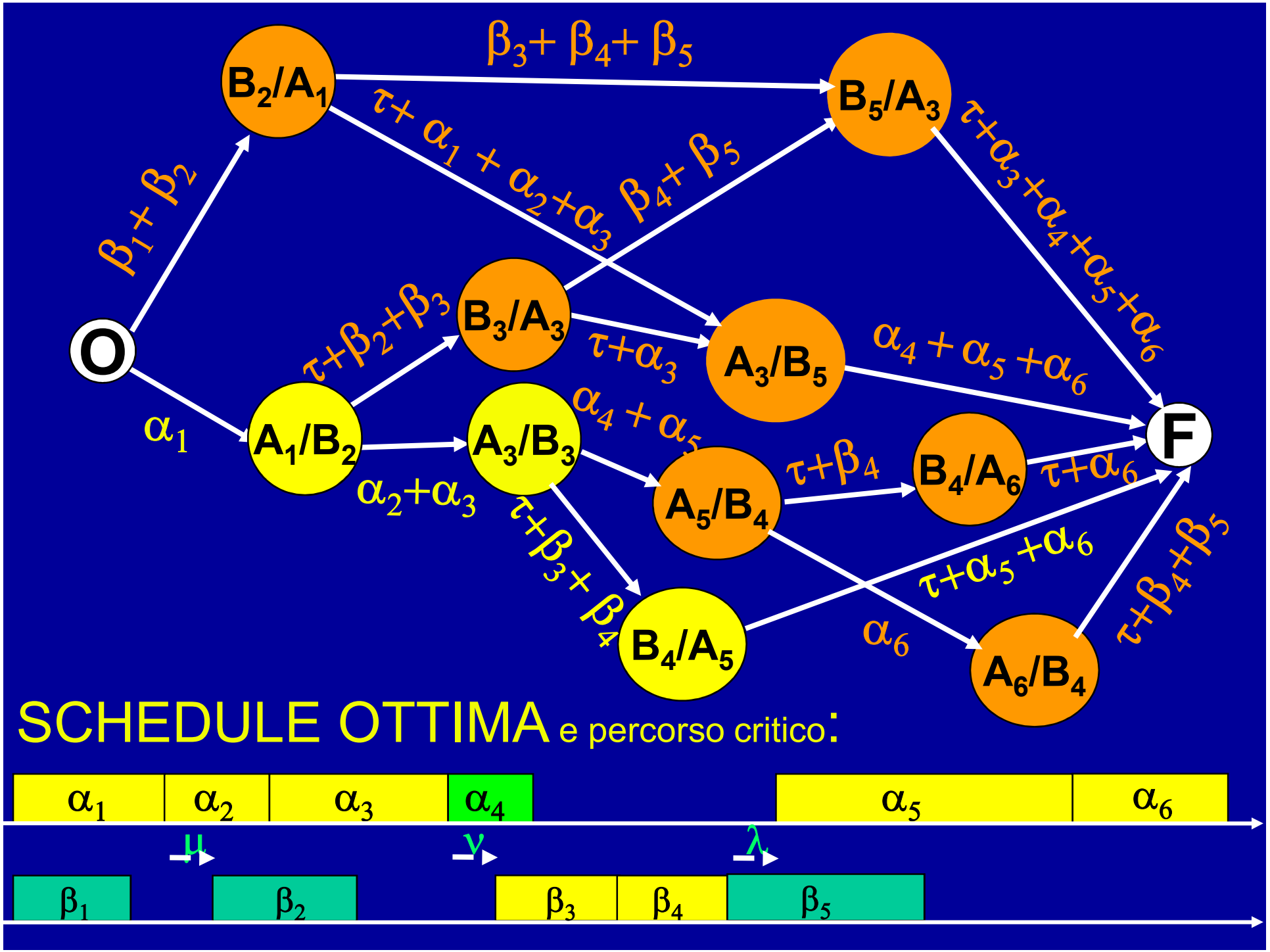
t.A	ut.A	t.B	ut.B	indici
α	cond.	β	cond.	op.
14	μ	12	ρ	1
10		16	μ	2
16	ν	14	ν	3
18	ρ	9	λ	4
31	λ	20	ν	5
14	λ			6

GRAFO COMPLETO

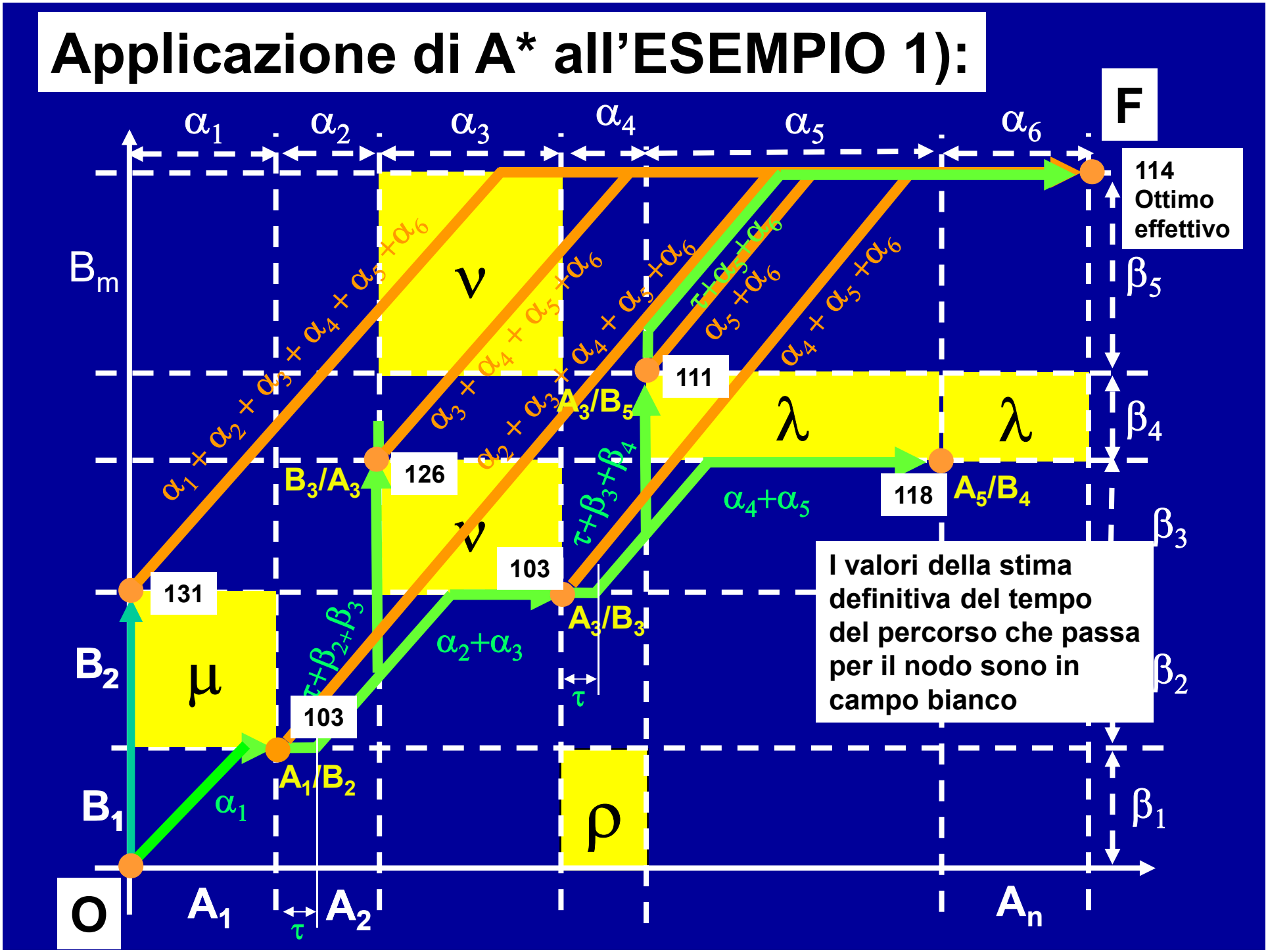
-Applicando Dijstra i nodi si sviluppano nell'ordine della numerazione (in nero i tempi candidati da O: il minimo è il definitivo)

ESEMPIO 1




O
B2/A1
B5/A3
B3/A3
A3/B5
A1/B2
A3/B3
A5/B4
B4/A6
F
B4/A5
A6/B4
β3+ β4+ β5
β1+β2
τ+ α1 + α2+α3
β4+ β5
τ+α3+α4+α5+α6
τ+β2+β3
τ+α3
α4 + α5 +α6
α1
α4 + α5
τ+β4
τ+α6
α2+α3
τ+β3+ β4
τ+α5 +α6
α6
τ+β4+β5
SCHEDULE OTTIMA e percorso critico:
α1
α2
α3
α4
α5
α6
μ
γ
λ
β1
β2
β3
β4
β5

Applicazione di A* all'ESEMPIO 1):

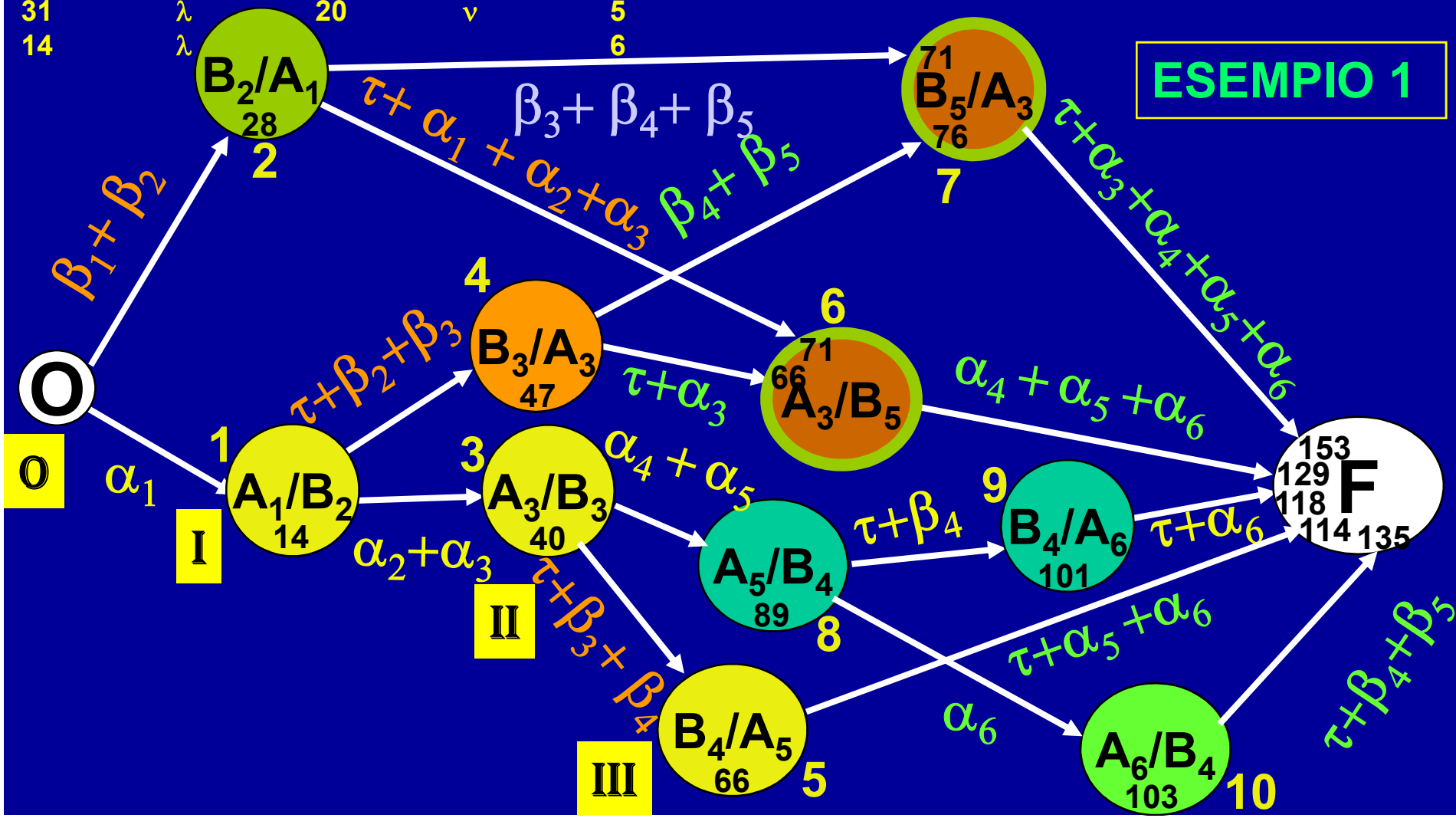


ESEMPIO 2JSP 1: Due lavori assegnati: A e B
Tempo di commutazione degli utensili $\tau = 3$

t.A	ut.A	t.B	ut.B	indici
α	cond.	β	cond.	op.
14	μ	12	ρ	1
10		16	μ	2
16	ν	14	ν	3
18	ρ	9	λ	4
31	λ	20	ν	5
14	λ			6

GRAFO COMPLETO
-Applicando Dijstra i nodi si sviluppano nell'ordine della numerazione (in nero i tempi candidati da O: il minimo è il definitivo)
- Con A* si sviluppano solo O e i nodi con i numeri romani

ESEMPIO 1



ESERCIZIO 1

Con i dati di cui sotto si calcoli il sequenziamento che minimizza C_m, utilizzando l'algoritmo A*

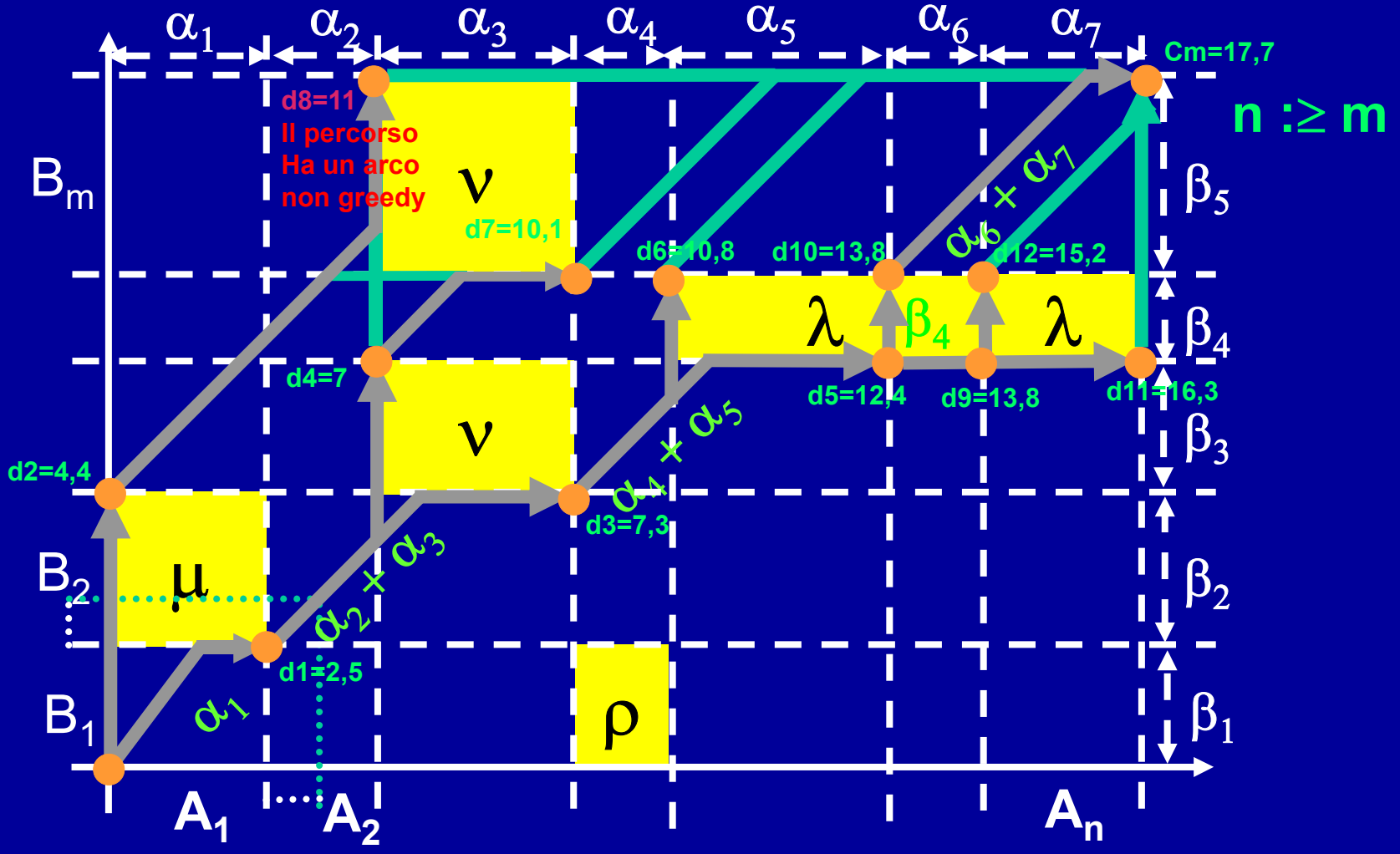
Due lavori assegnati: A e B

Tempo di commutazione degli utensili τ = 0,75

t.A	ut.A	t.B	ut.B		indici
α	cond.	β	cond.	op.	
2,5	μ		2	ρ	1
1,75			2,5	μ	2
3	ν		2	ν	3
1,5	ρ		1,5	λ	4
3,5	λ		3	ν	5
1,5	λ				6
2,5	λ				7

Programmazione dinamica per il calcolo percorso minimo: min Cm

Si calcolano i di (distanza minima dall'origine del nodo i) trovando i percorsi minimi d fino a i: → di



in questo caso il sequenziamento greedy minimizza Cm

*ESEMPIO 3): UNA APPLICAZIONE DI A**

Problema

Un flusso continuo di pezzi subisce ciascuno una sequenza di 10 operazioni: su una prima macchina le operazioni da O_1 a O_{v-1} e poi su una seconda macchina da O_v a O_{10}, senza buffer fra le due macchine.

Flusso continuo senza buffer significa che la prima macchina esegue la sottosequenza da O_1 a O_{v-1} nello stesso periodo di tempo in cui la seconda macchina esegue la sottosequenza da O_v a O_{10}.

UNA APPLICAZIONE DI A*

Problema

Le operazioni richiedono i seguenti tempi:

Operazioni	1	2	3	4	5	6	7	8	9	10
Tempi	3	2	1	4	5	6	4	3	2	1

Tra le operazioni vi sono le seguenti incompatibilità, dovute alla condivisione di risorse terze, indicate con lettere maiuscole: $(O_1O_3O_6)_A$; $(O_2O_7)_B$; $(O_4O_7)_C$; $(O_6O_8)_D$.

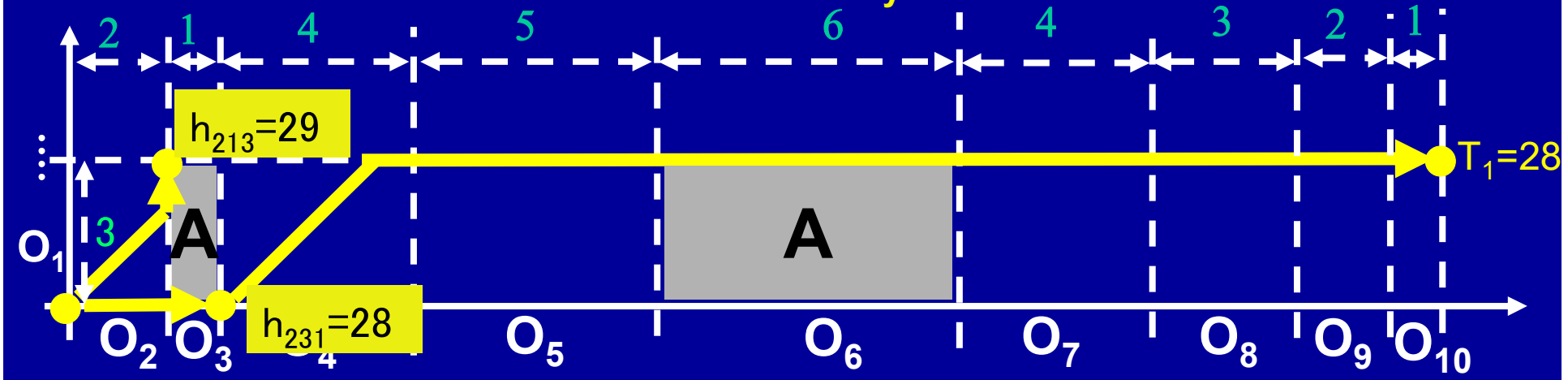
Utilizzando il grafo generato dalla griglia di condivisione delle risorse terze, si scelga v (ovviamente da 2 a 10) in modo da minimizzare il tempo di ciclo, cioè di esecuzione in parallelo delle due sottosequenze.

Si noti che, invece di usare 9 griglie, una per ciascun valore di v, che generano 9 grafi con un'origine e una destinazione, si può usare una sola griglia per generare i 9 grafi, ciascuno con la sua origine e la sua destinazione.

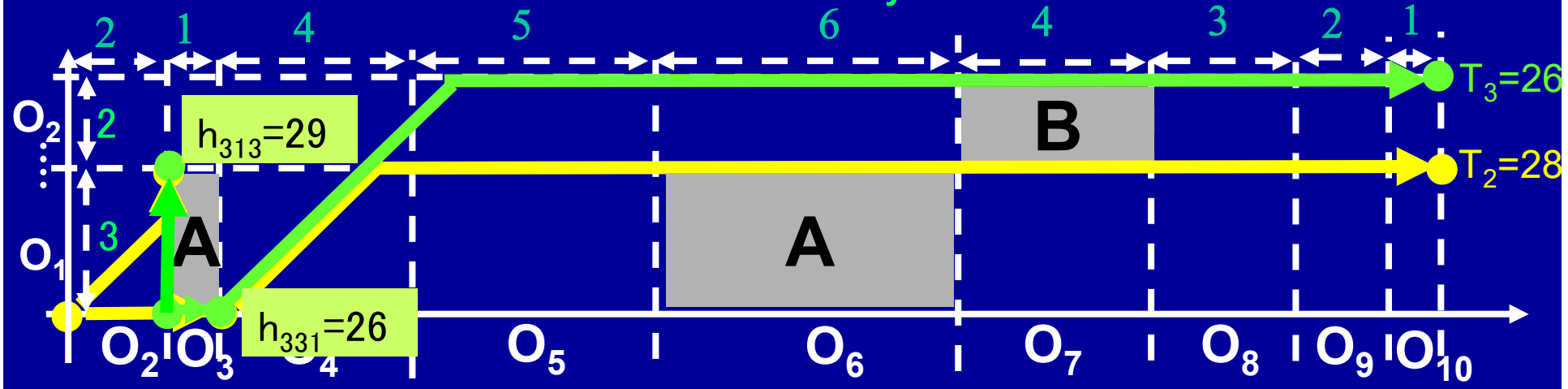
UNA APPLICAZIONE DI A*

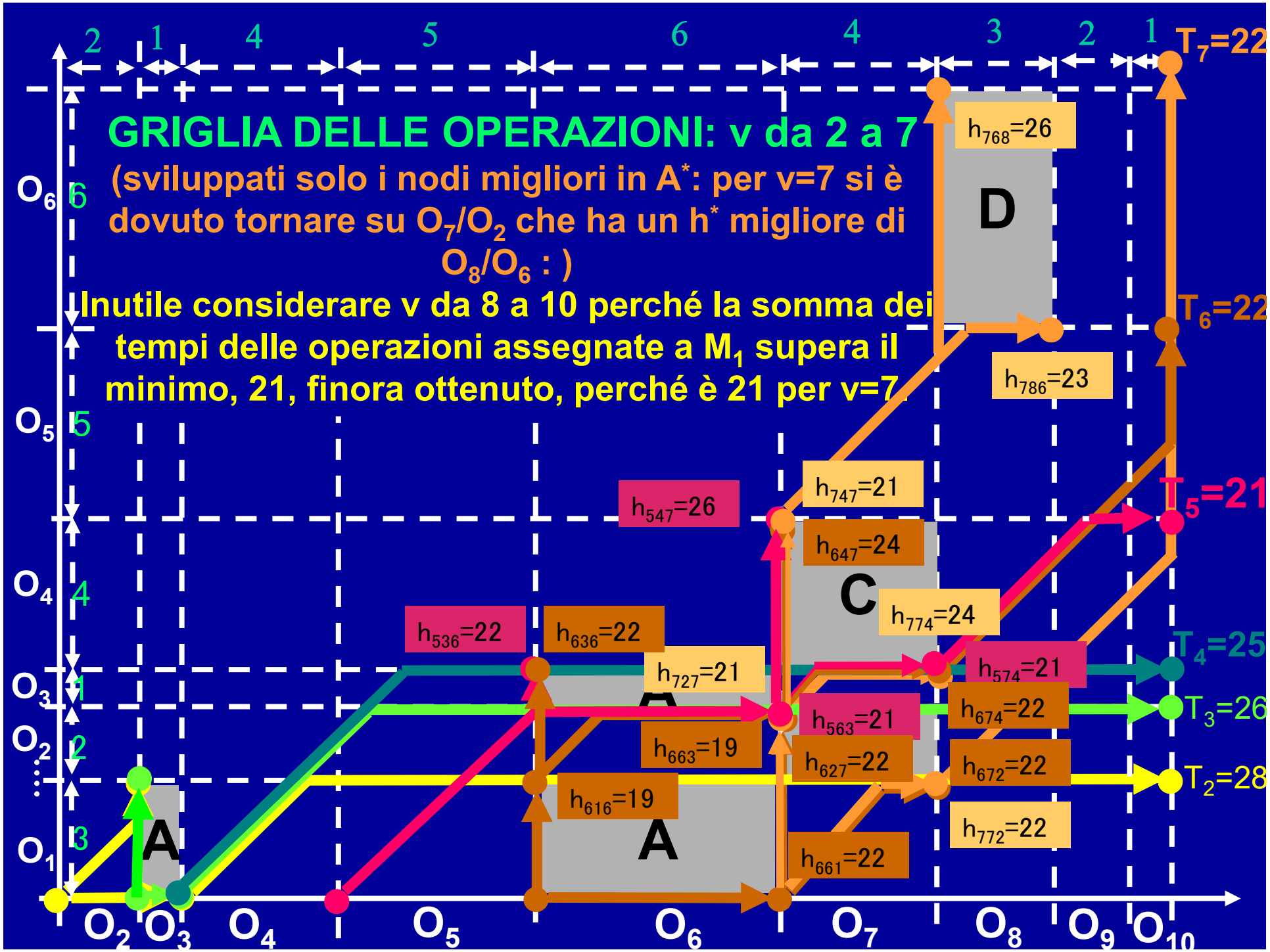
h_{ixy}: stima del tempo minimo di un percorso che inizia con O_i e passa per O_x/O_y

GRIGLIA DELLE OPERAZIONI e h_{2xy}: v = 2



GRIGLIA DELLE OPERAZIONI e h_{3xy}: v = 3




GRIGLIA DELLE OPERAZIONI: v da 2 a 7
(sviluppati solo i nodi migliori in A*: per v=7 si è dovuto tornare su O7/O2 che ha un h* migliore di O8/O6 :)
Inutile considerare v da 8 a 10 perché la somma dei tempi delle operazioni assegnate a M1 supera il minimo, 21, finora ottenuto, perché è 21 per v=7
2 1 4 5 6 4 3 2 1
T7=22
T6=22
T5=21
T4=25
T3=26
T2=28
O6 6
O5 5
O4 4
O3 1
O2 2
O1 3
O2 O3 O4 O5 O6 O7 O8 O9 O10
h768=26
h786=23
h747=21
h547=26
h647=24
h774=24
h536=22
h636=22
h727=21
h574=21
h674=22
h563=21
h663=19
h627=22
h672=22
h616=19
h772=22
h661=22
D
C
A
A
A

ESERCIZIO 2: Si risolva lo stesso problema di cui nei quadri precedenti, con le operazioni di cui sono riportati nella prima colonna gl'indici e nella seconda tempi, con la sola incompatibilità tra O_2, O_4 e O_5, dovuta all'utilizzo dello stesso utensile α

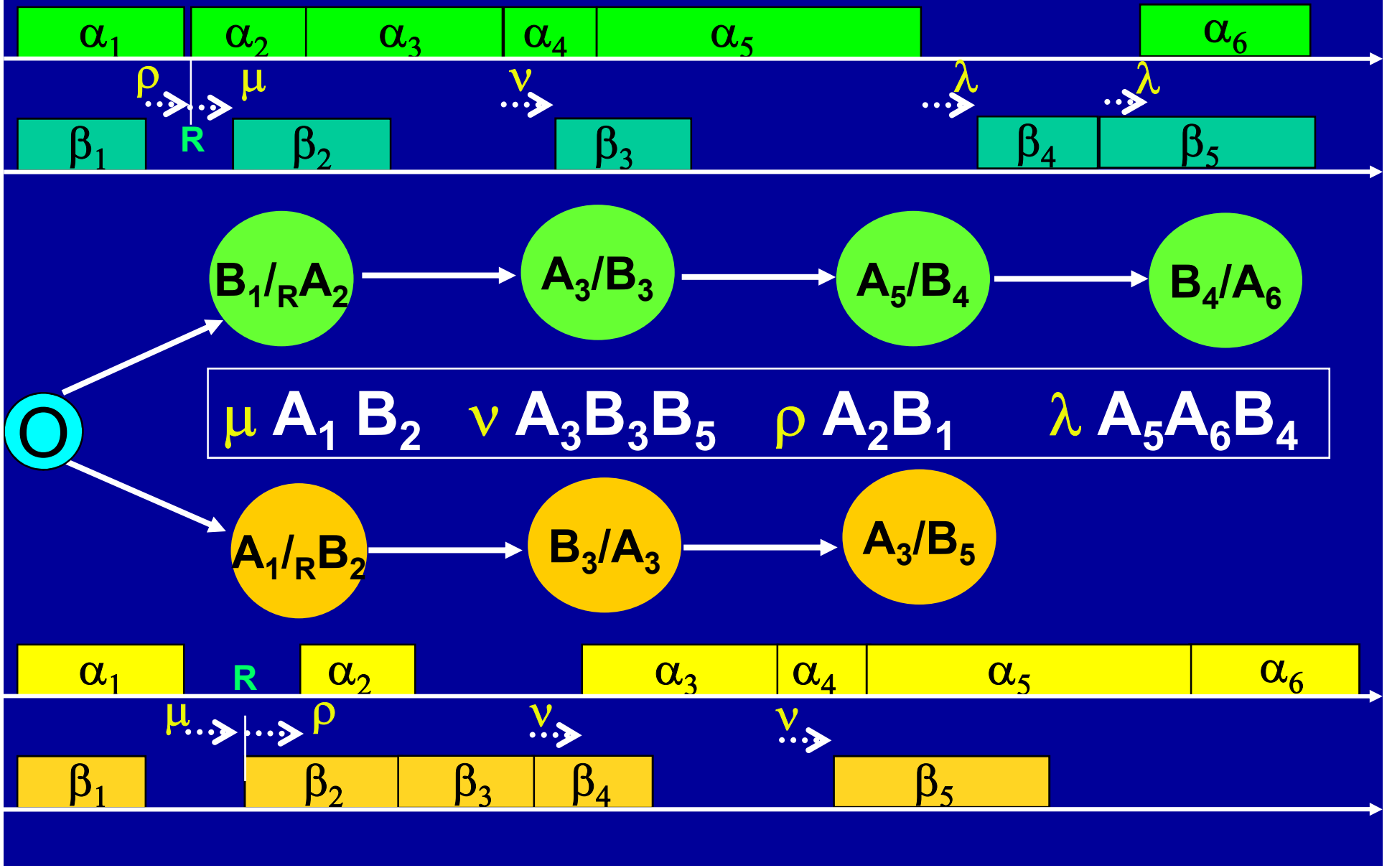
i	t	u
1	3	
2	2	α
3	1	
4	4	α
5	2	α

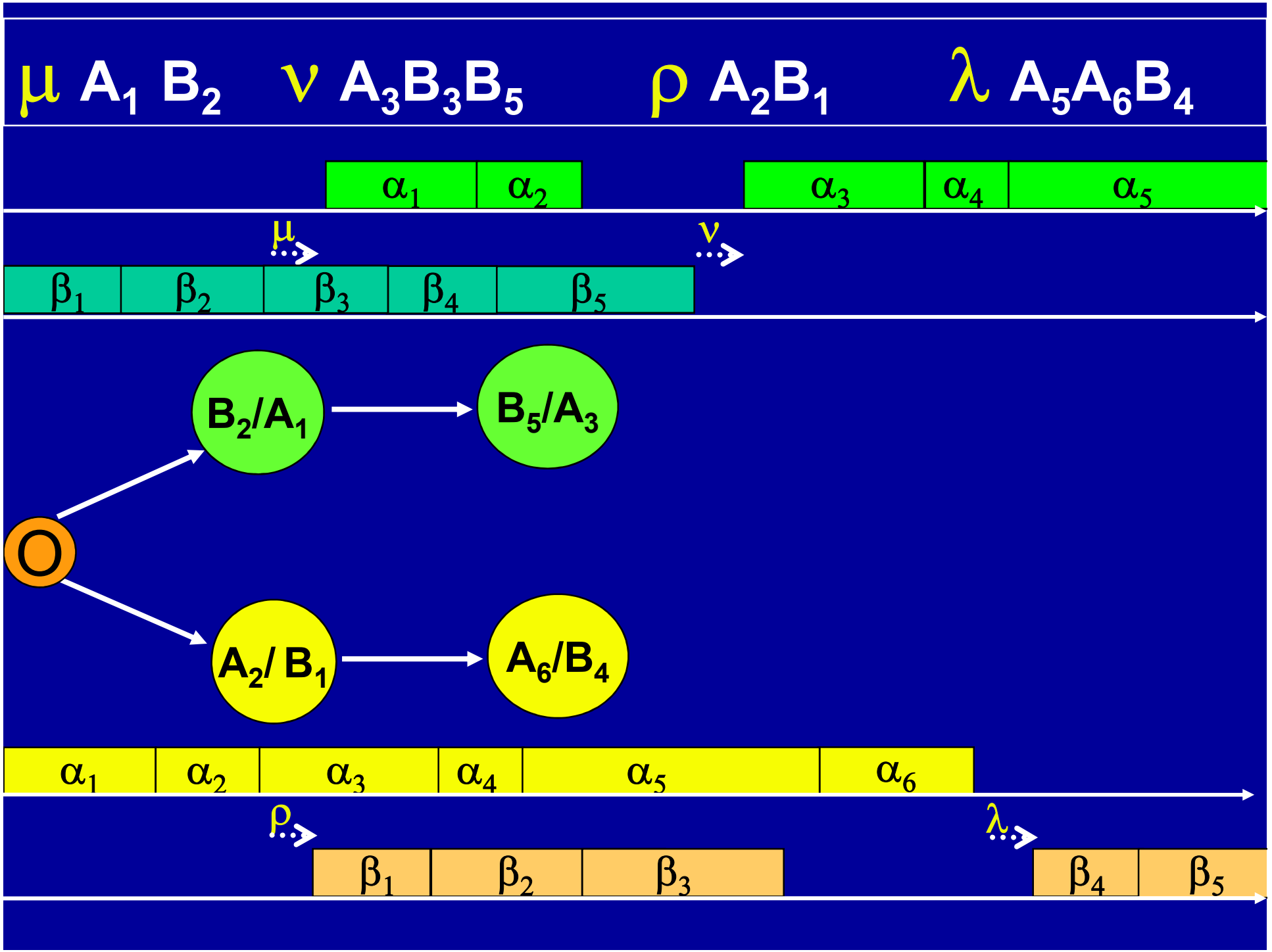
Con un tempo di trasporto dell'utensile α:

$$\tau = 0 \text{ oppure } 0.5$$

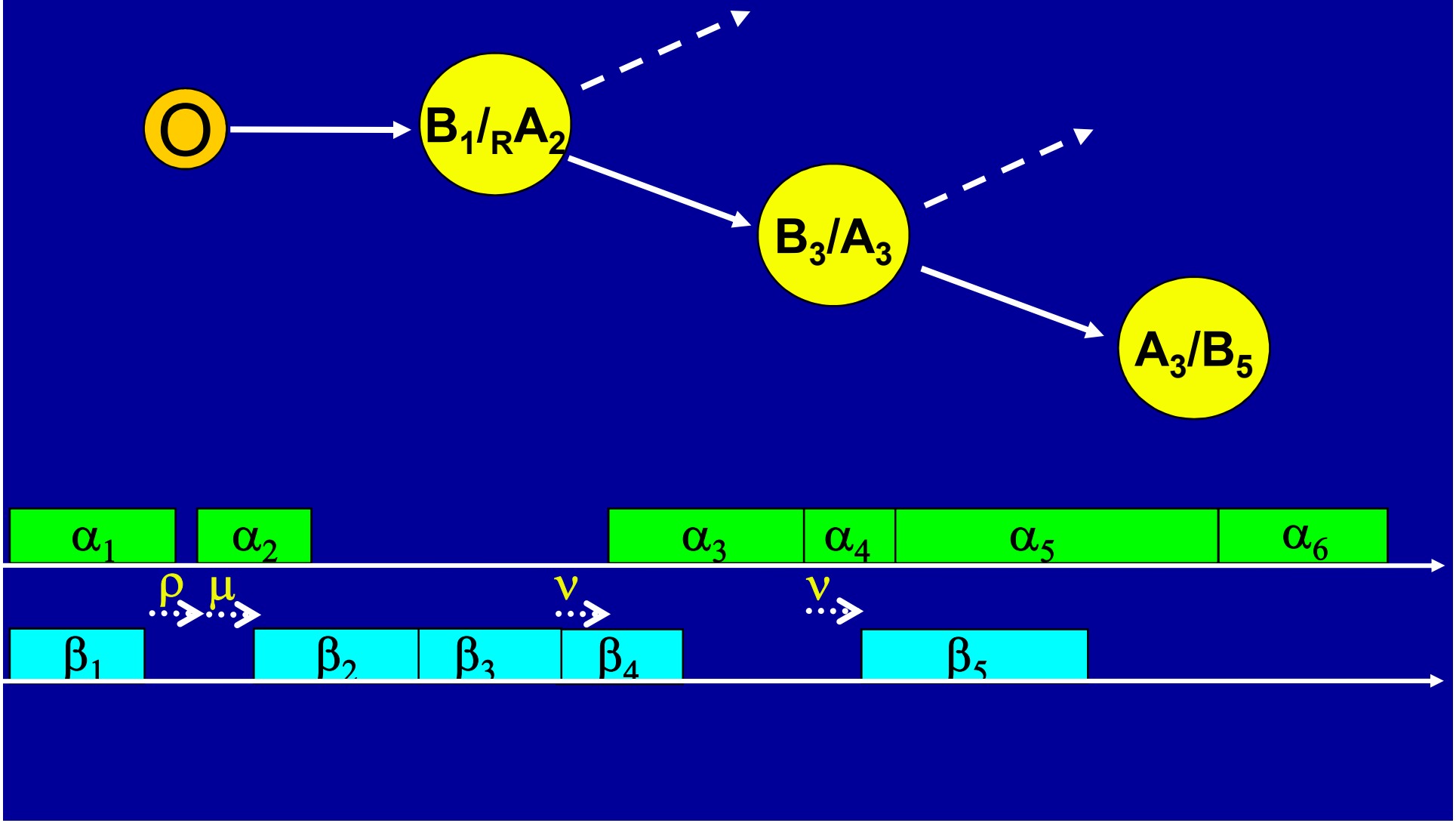
2JSP: CONFLITTO DOPPIO

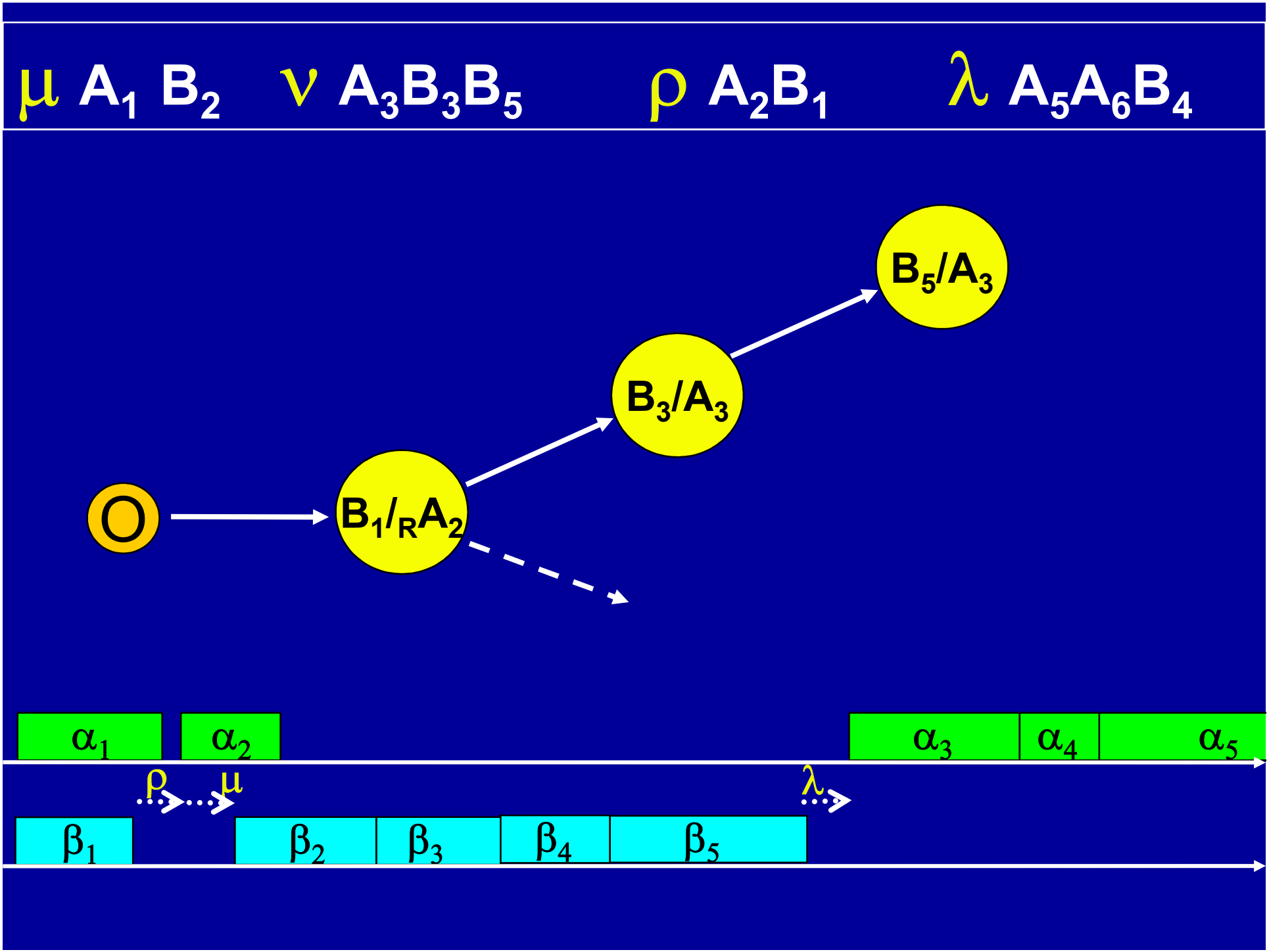
(Si veda anche Agnetis ”Dispense …”: 2.5 Gestione dinamica ... pp. 69_ _90)



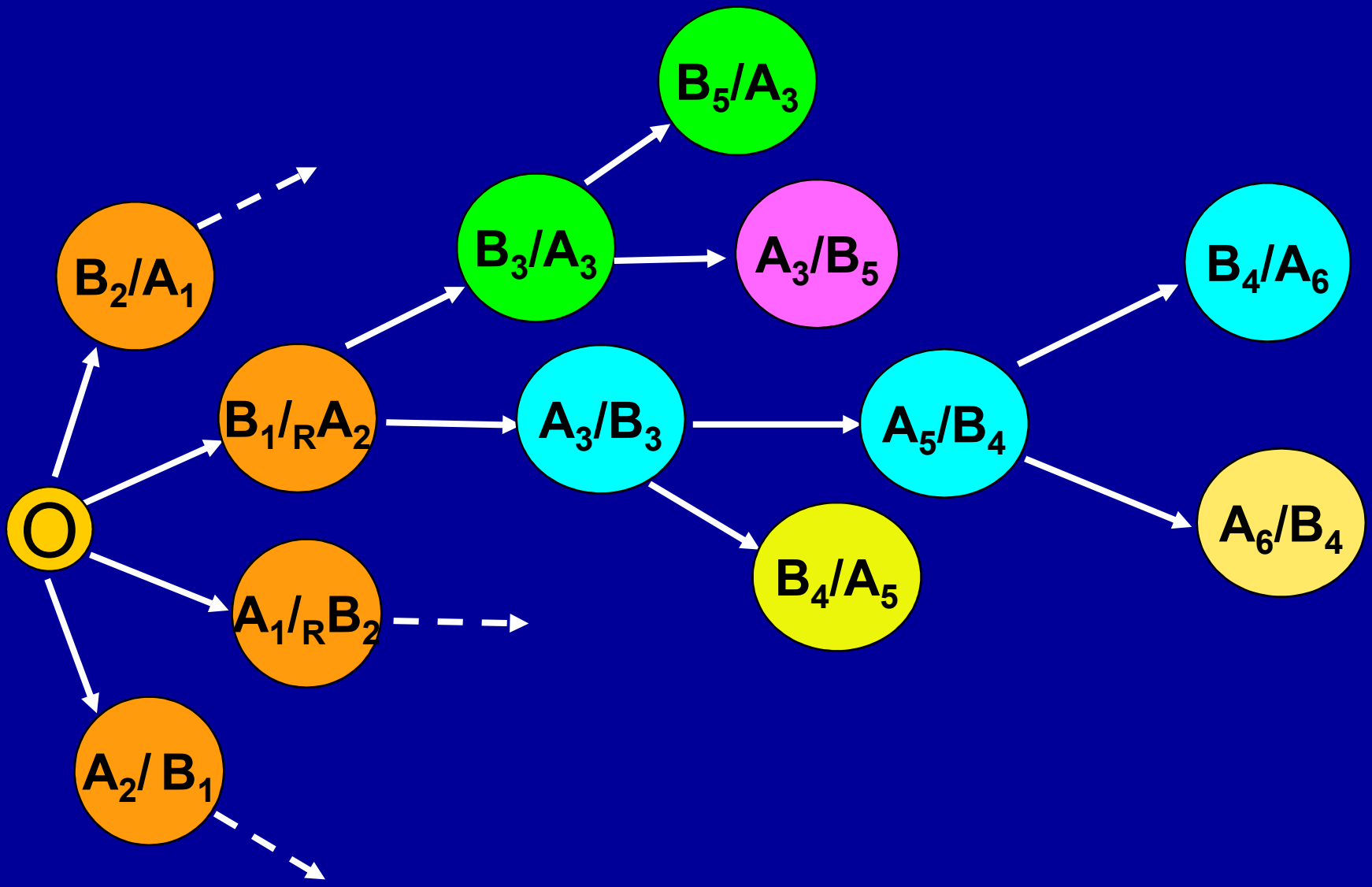

μ A1 B2
ν A3B3B5
ρ A2B1
λ A5A6B4
α1
α2
α3
α4
α5
μ
ν
β1
β2
β3
β4
β5
B2/A1
B5/A3
O
A2/ B1
A6/B4
α1
α2
α3
α4
α5
α6
ρ
λ
β1
β2
β3
β4
β5

μ A_1 B_2 ν $A_3B_3B_5$ ρ A_2B_1 λ $A_5A_6B_4$



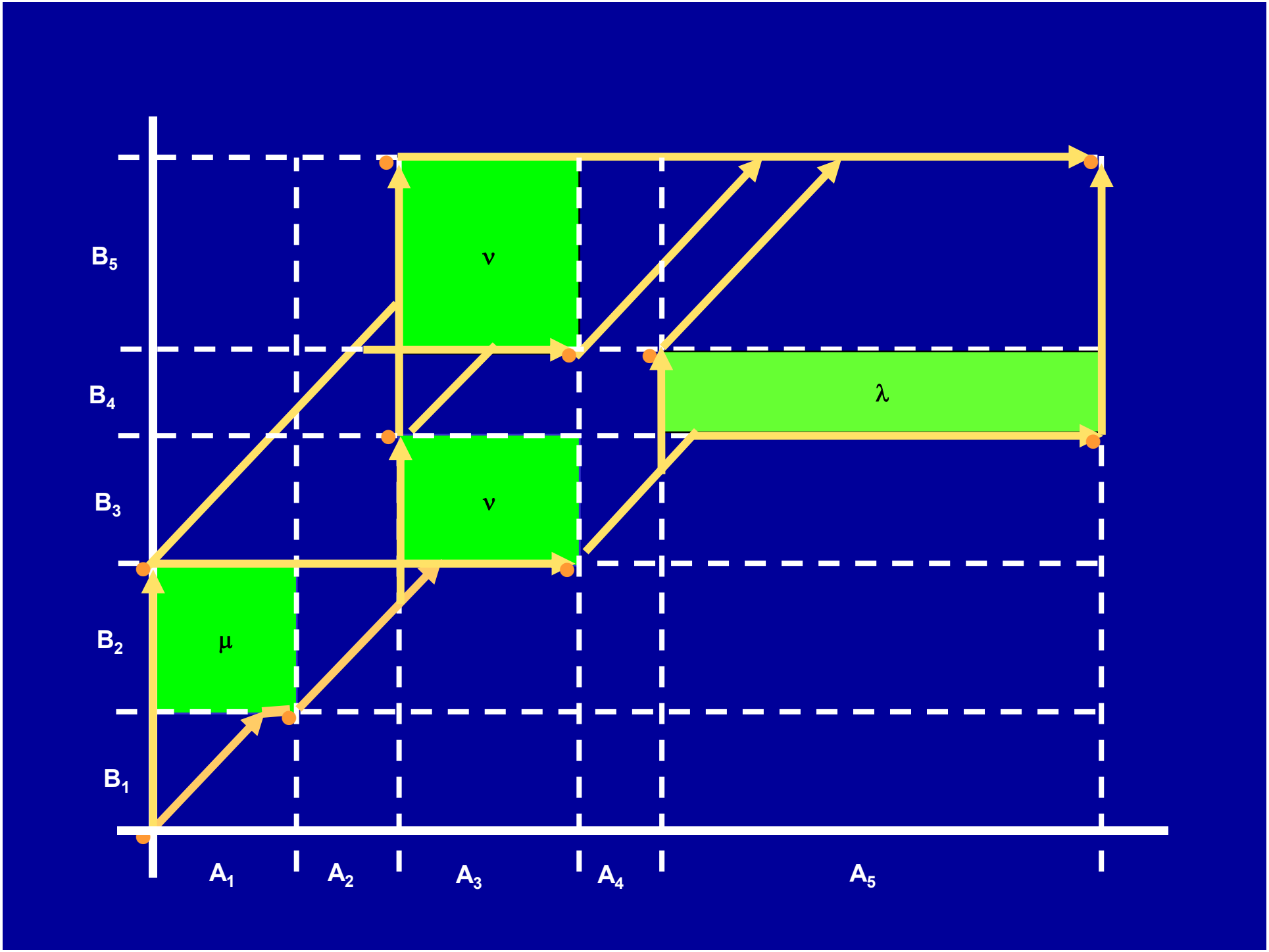

μ A1 B2
ν A3B3B5
ρ A2B1
λ A5A6B4
B5/A3
B3/A3
O
B1/RA2
α1
α2
α3
α4
α5
ρ
μ
λ
β1
β2
β3
β4
β5

μ A_1 B_2 ν $A_3B_3B_5$ ρ A_2B_1 λ $A_5A_6B_4$

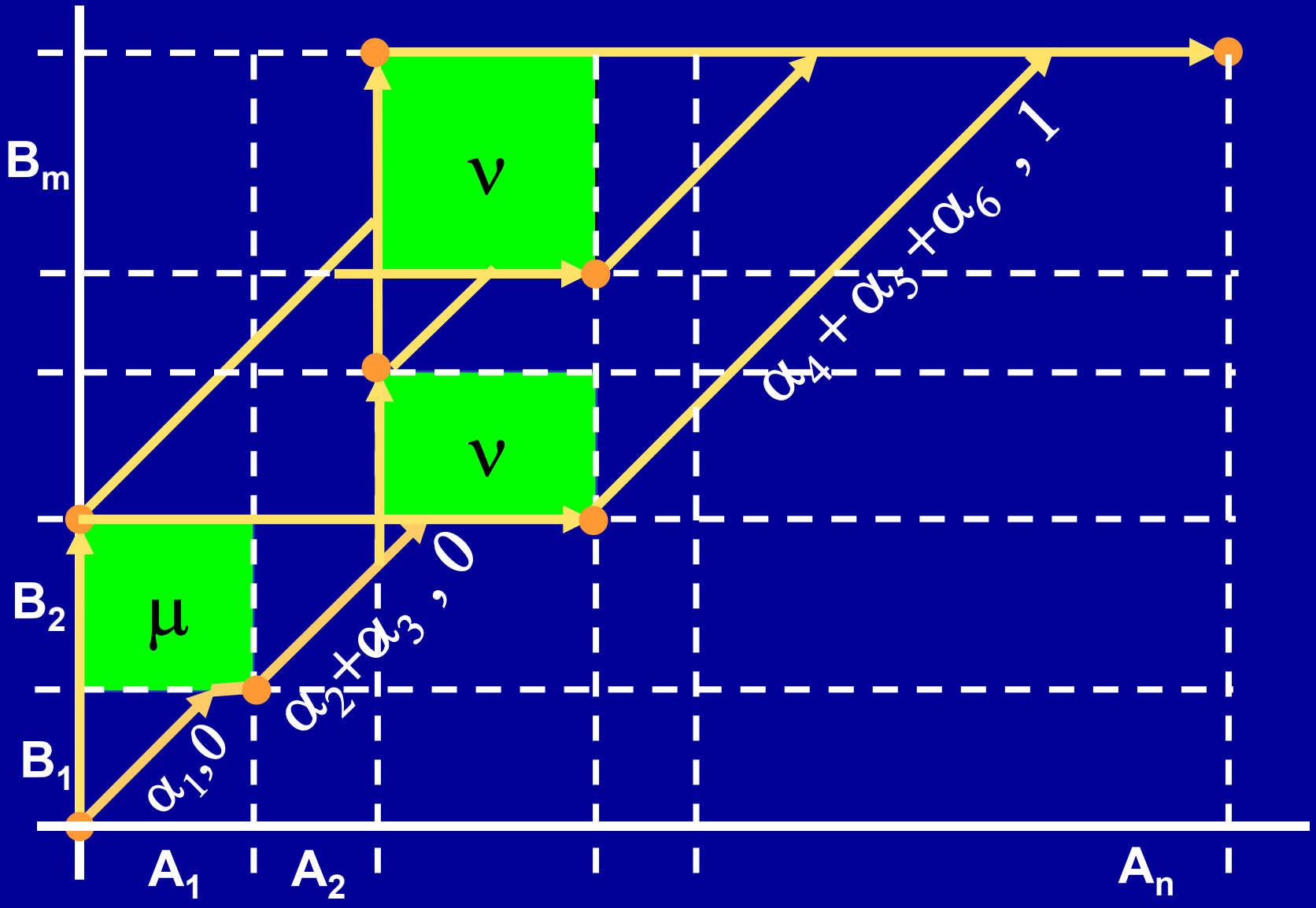


DUPLICAZIONE DEGLI UTENSILI

La duplicazione degli utensili riduce i conflitti, duplicare un utensile vuol dire eliminare alcune zone critiche


B5
B4
B3
B2
B1
A1
A2
A3
A4
A5
ν
ν
λ
μ

UTENSILE λ DUPLICATO



il problema diviene

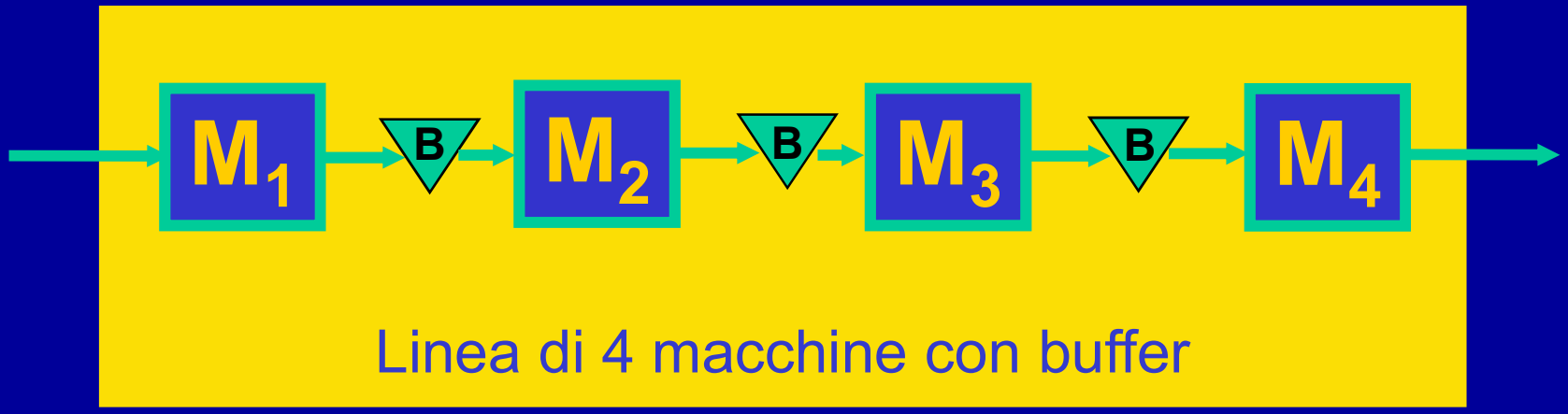
quindi duplice:

- minimizzazione del tempo di completamento
- minimizzazione del numero di utensili duplicati

si può procedere come per la minimizzazione di $\underline{F}$ ed T_{max}:

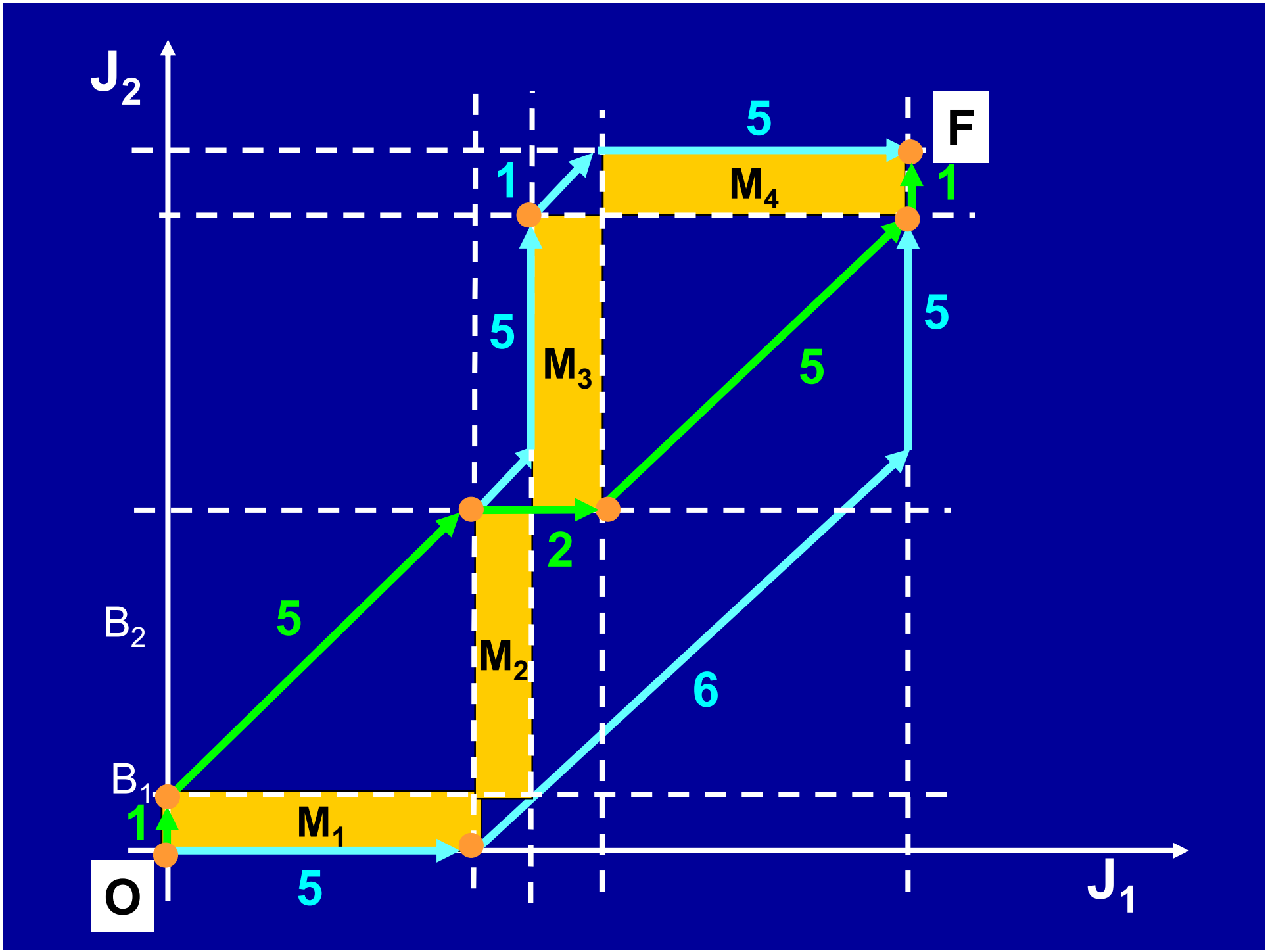
si minimizza il tempo con un vincolo sulla duplicazione, per trovare poi una curva di efficienza muovendo il vincolo

La soluzione del 2JSP è la generalizzazione di quella data da Aker per 2 lavori $\mathbf{J_i}$ di m operazioni in serie da eseguire su una linea con buffer intermedi: ciascuno di lunghezza temporale t_i sulla macchina M_i (i=1,…, m)

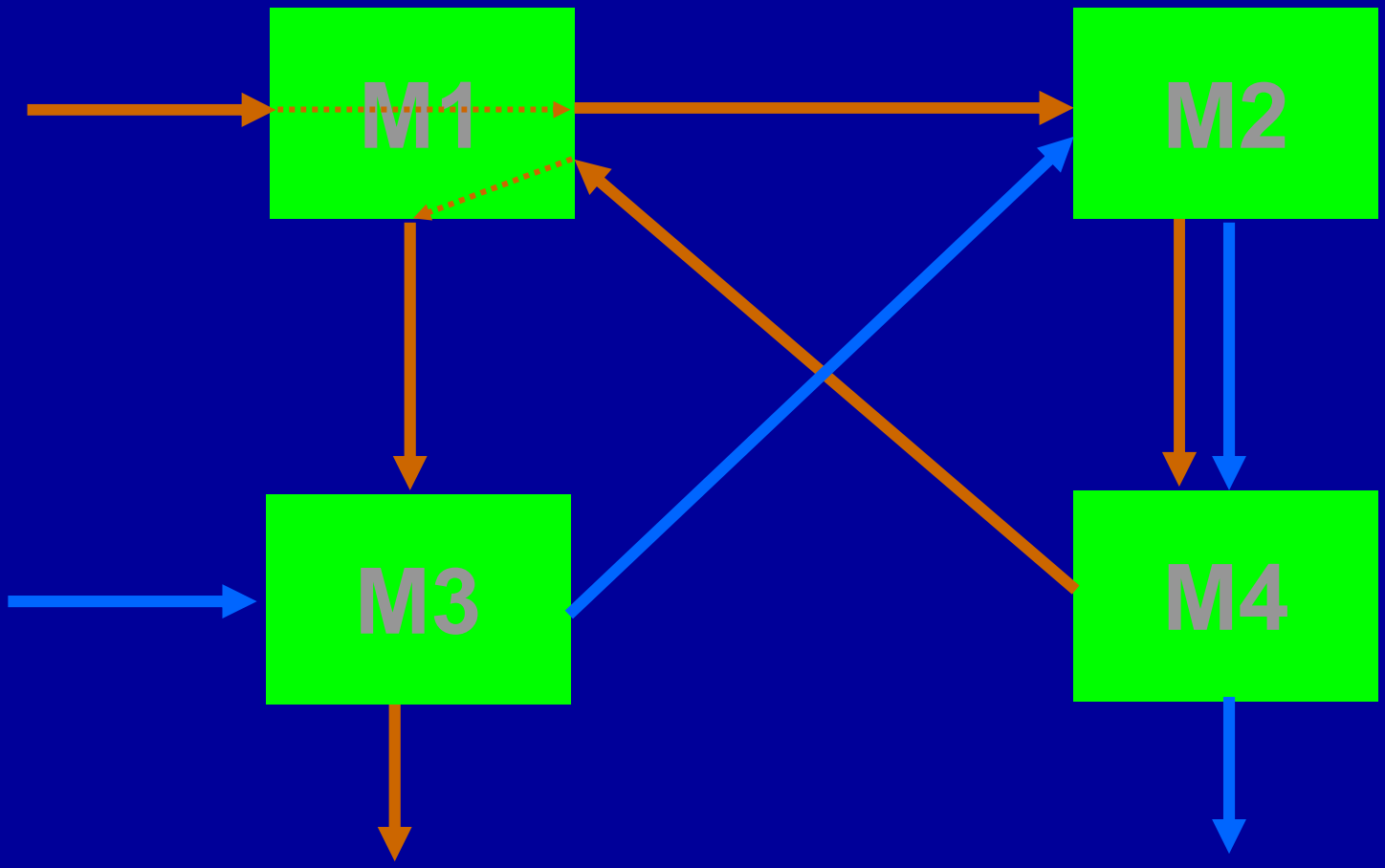


Linea di 4 macchine con buffer

Per minimizzare il tempo di completamento C_m si usa una griglia di condivisione in cui le zone proibite riguardano le macchine


J2
J1
O
F
B2
B1
M1
M2
M3
M4
5
1
5
1
5
5
2
5
6
5

Min C_m si può risolvere come un 2JSP allo stesso modo



Min C_m non si può risolvere allo stesso modo quando i lavori sono n>2 (già per n=3 le zone proibite sarebbero cubi e la progressione greedy potrebbe incidere su una faccia ….)