Composizione di uno *stemma codicum* con TikZ

Gianluca Pignalberi

Napoli, 27 ottobre 2012

Sommario

Sommario

Sebbene LaTeX abbia ottimi pacchetti dedicati alla composizione di uno *stemma codicum* più o meno complesso, TikZ fa ottenere risultati almeno altrettanto buoni senza bisogno di introdurre maggior complessità notazionale.

Abstract

While LaTeX has very good packages to typeset a more or less complex *stemma codicum*, TikZ allows to reach results at least as good without requiring harder notational complexity.

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.
Formato da:

nodi: simboli con didascalia

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.
Formato da:

nodi: simboli con didascalia

archi: segmenti che uniscono due nodi a livelli adiacenti

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.
Formato da:

- nodi: simboli con didascalia
- archi: segmenti che uniscono due nodi a livelli adiacenti

Ogni nodo può avere un solo genitore

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.
Formato da:

nodi: simboli con didascalia

archi: segmenti che uniscono due nodi a livelli adiacenti

Ogni nodo può avere un solo genitore
La *radice* non ha genitori,

Alberi semplici

Albero

Struttura per rappresentare strutture gerarchiche.
Formato da:

nodi: simboli con didascalia

archi: segmenti che uniscono due nodi a livelli adiacenti

Ogni nodo può avere un solo genitore
La *radice* non ha genitori, le *foglie* non hanno figli

Alberi semplici

Albero

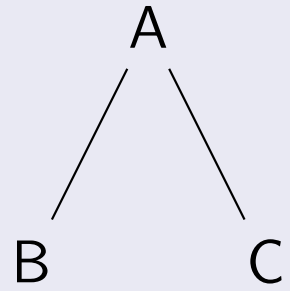
Struttura per rappresentare strutture gerarchiche.
Formato da:

nodi: simboli con didascalia

archi: segmenti che uniscono due nodi a livelli adiacenti

Ogni nodo può avere un solo genitore
La *radice* non ha genitori, le *foglie* non hanno figli

Esempio



Alberi semplici

Albero

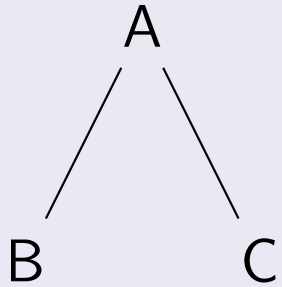
Struttura per rappresentare strutture gerarchiche.
Formato da:

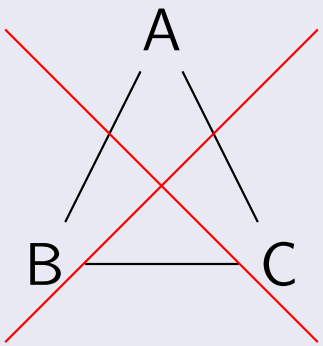
nodi: simboli con didascalia

archi: segmenti che uniscono due nodi a livelli adiacenti

Ogni nodo può avere un solo genitore
La *radice* non ha genitori, le *foglie* non hanno figli

Esempio





Alberi semplici

Codice dell'albero

```
\begin{tikzpicture}
\node{A}
  child{node{B}}
  child{node{C}}
;
\end{tikzpicture}
```

Alberi affiancati

Due alberi separati vengono disegnati sovrapposti

Alberi affiancati

Due alberi separati vengono disegnati sovrapposti
Dobbiamo dare una posizione agli alberi successivi al primo

Alberi affiancati

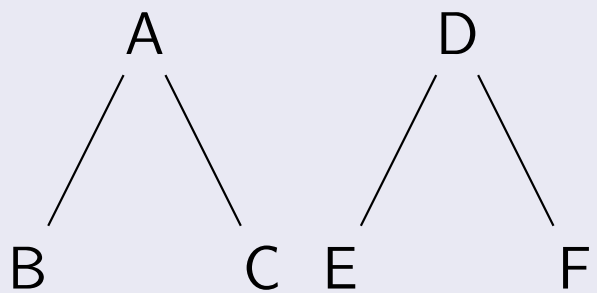
Due alberi separati vengono disegnati sovrapposti
Dobbiamo dare una posizione agli alberi successivi al primo

Codice di alberi affiancati

```
\begin{tikzpicture}
\node{A}
  child{node{B}}
  child{node{C}}
;
\node at (2,0) {D}
  child{node{E}}
  child{node{F}}
;
\end{tikzpicture}
```

Alberi affiancati

Due alberi affiancati



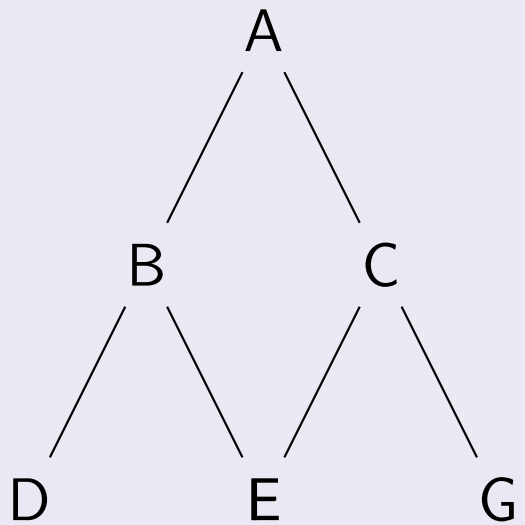
Alberi a piú livelli

Codice di un albero a tre livelli

```
\begin{tikzpicture}
\node{A}
  child{node{B}
    child{node{D}}
    child{node{E}}}
  child{node{C}
    child{node{F}}
    child{node{G}}}
;
\end{tikzpicture}
```

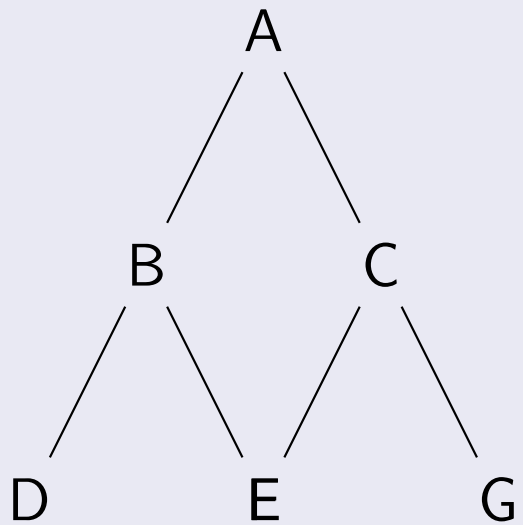
Alberi a piú livelli

Albero a tre livelli



Alberi a piú livelli

Albero a tre livelli



Problema

I due figli piú interni sono sovrapposti. Bisogna correggere!

Alberi a piú livelli

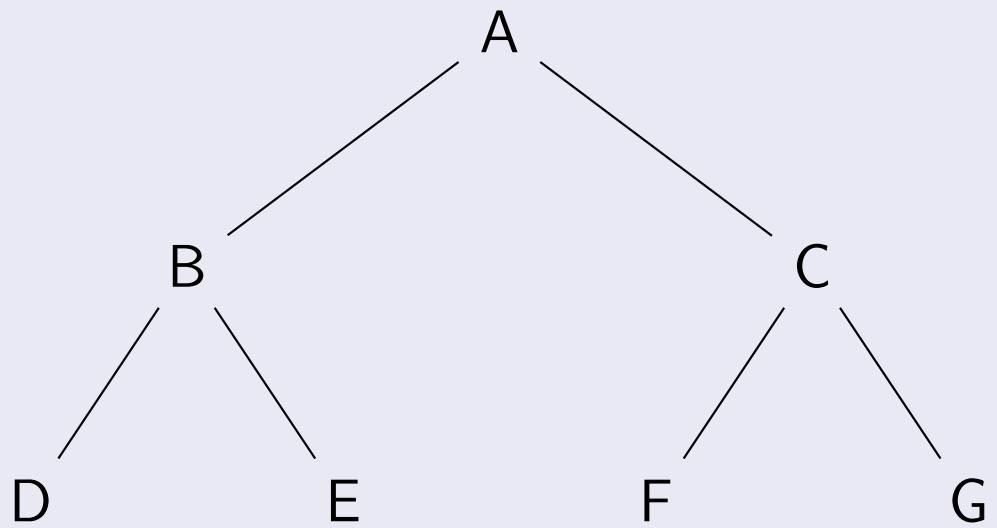
Usiamo l'attributo `sibling distance`

Uso di `sibling distance`

```
\begin{tikzpicture}
[level 1/.style={sibling distance=40mm},
 level 2/.style={sibling distance=20mm}]
\node{A}
  child{node{B}
    child{node{D}}
    child{node{E}}}
  child{node{C}
    child{node{F}}
    child{node{G}}}
;
\end{tikzpicture}
```

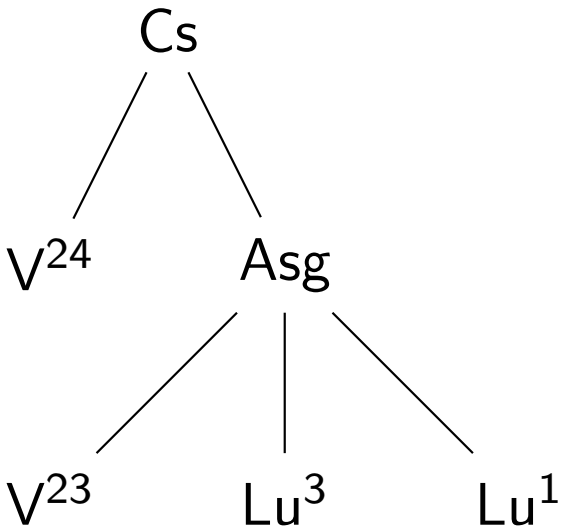
Alberi a piú livelli

Risultato

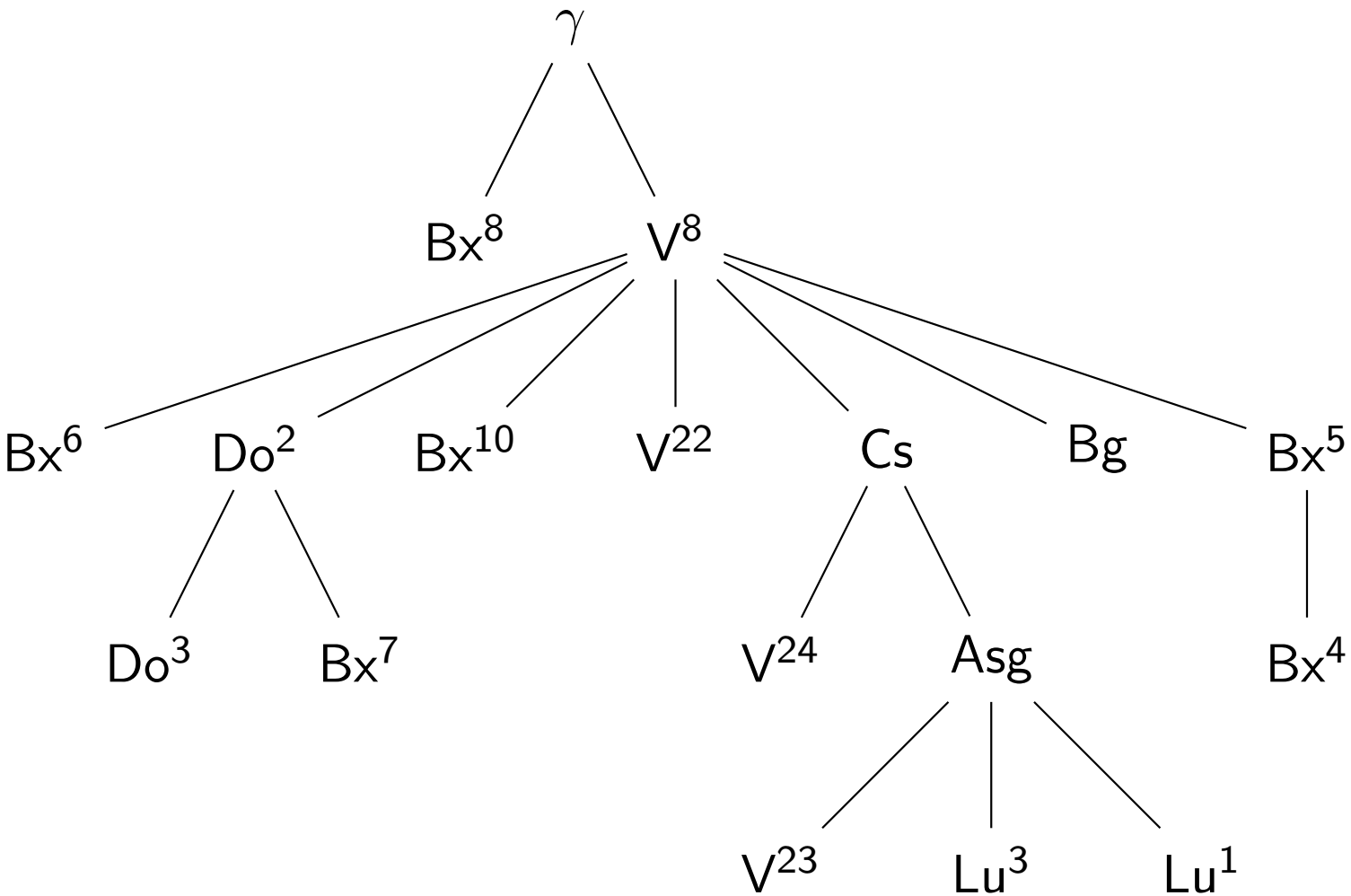


Stemmata codicum concreti

```
\begin{tikzpicture}
\node {Cs}
    child { node {V$^{24}$} }
    child { node {Asg}
      child { node {V$^{23}$} }
      child { node {Lu$^{3}$} }
      child { node {Lu$^{1}$} }
    };
\end{tikzpicture}
```



Stemmata codicum concreti



Distanza tra i livelli

Come possiamo variare la distanza tra i nodi dello stesso livello, cosí possiamo variare la distanza tra due livelli adiacenti

Distanza tra i livelli

Come possiamo variare la distanza tra i nodi dello stesso livello, cosí possiamo variare la distanza tra due livelli adiacenti
L'attributo è `level distance`

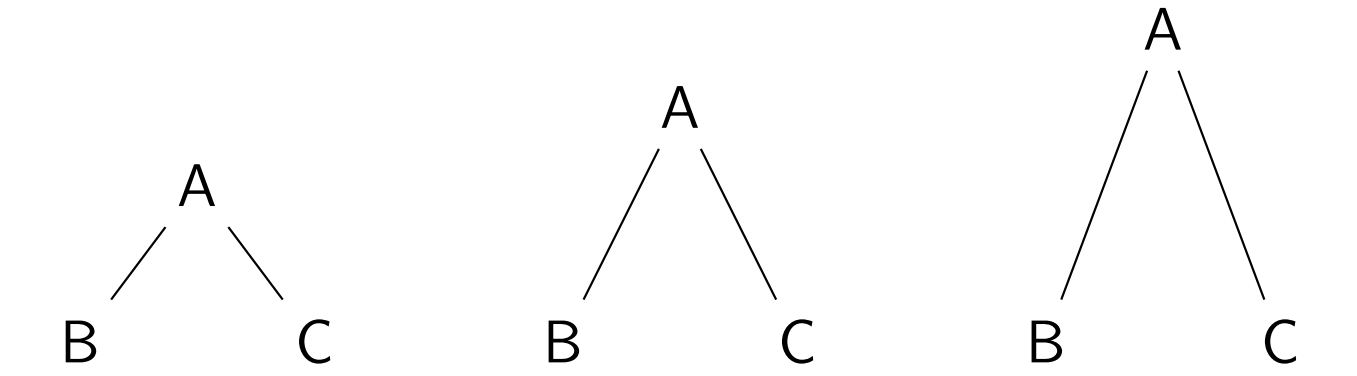
Distanza tra i livelli

Come possiamo variare la distanza tra i nodi dello stesso livello, cosí possiamo variare la distanza tra due livelli adiacenti
L'attributo è `level distance`

Esempio

```
\begin{tikzpicture}[level 1/.style={level distance=3cm}]
```

Distanza tra i livelli



Archi arbitrari

Per disegnare degli archi arbitrari tra nodi arbitrari dobbiamo per prima cosa etichettare i nodi da collegare:

Archi arbitrari

Per disegnare degli archi arbitrari tra nodi arbitrari dobbiamo per prima cosa etichettare i nodi da collegare:

```
\node (g) {$\gamma$}
  child { node (E) {Bx$^{8}$} }
```

Archi arbitrari

Per disegnare degli archi arbitrari tra nodi arbitrari dobbiamo per prima cosa etichettare i nodi da collegare:

```
\node (g) {$\gamma$}
  child { node (E) {Bx$^{8}$} }
```

Quindi disegniamo l'arco corrispondente:

Archi arbitrari

Per disegnare degli archi arbitrari tra nodi arbitrari dobbiamo per prima cosa etichettare i nodi da collegare:

```
\node (g) {$\gamma$}
  child { node (E) {Bx$^{8}$} }
```

Quindi disegniamo l'arco corrispondente:

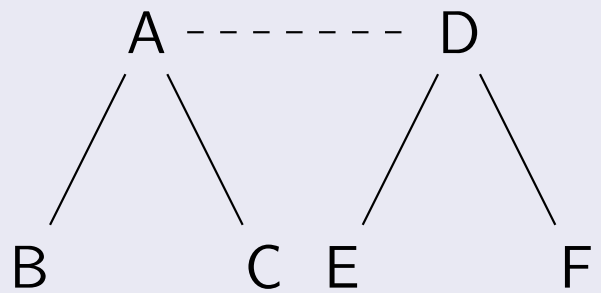
```
\draw[dashed,-] (g) -- (E);
```

Collegamento di alberi separati

```
\begin{tikzpicture}
\node(A){A}
  child{node{B}}
  child{node{C}}
;
\node at (2,0) (D){D}
  child{node{E}}
  child{node{F}}
;
\draw [dashed] (A) -- (D);
\end{tikzpicture}
```

Collegamento di alberi separati

Alberi separati uniti per le radici



Collegamento di nodi arbitrari: caso reale

```
\begin{tikzpicture}
\node (g) {$\gamma$}
  child { node (E) {Bx$^{8}$} }
  child { node {V$^{8}$}
    child { node {Bx$^{6}$} }
    child { node {Do$^{2}$}
      child { node {Do$^{3}$} }
      child { node (B) {Bx$^{7}$} }
    }
```

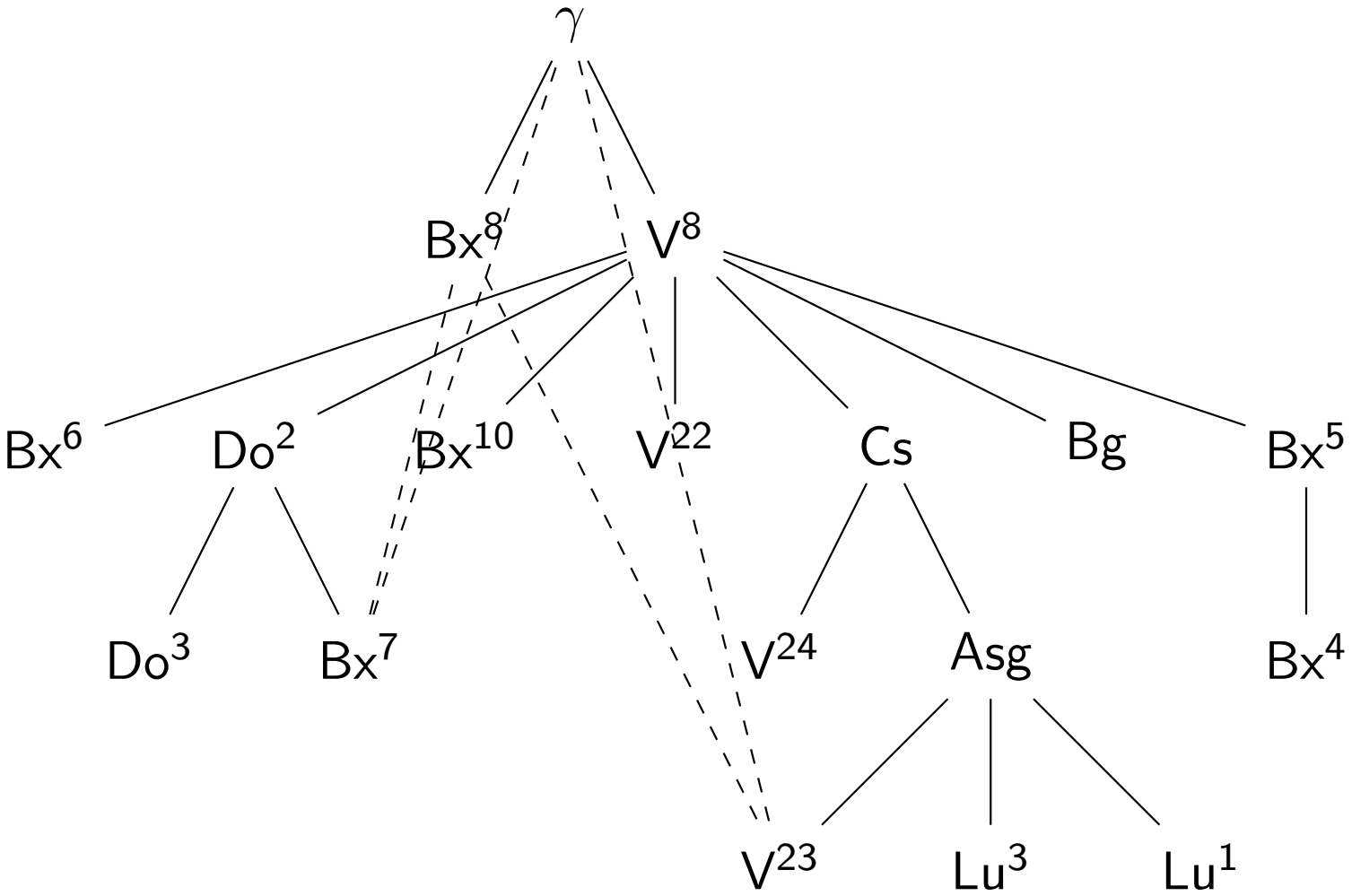
Collegamento di nodi arbitrari: caso reale

```
child { node {Bx$^{10}$} }
child { node {V$^{22}$} }
child { node {Cs}
  child { node {V$^{24}$} }
  child { node {Asg}
    child { node (v23) {V$^{23}$} }
    child { node {Lu$^{3}$} }
    child { node {Lu$^{1}$} }
  }
}
```

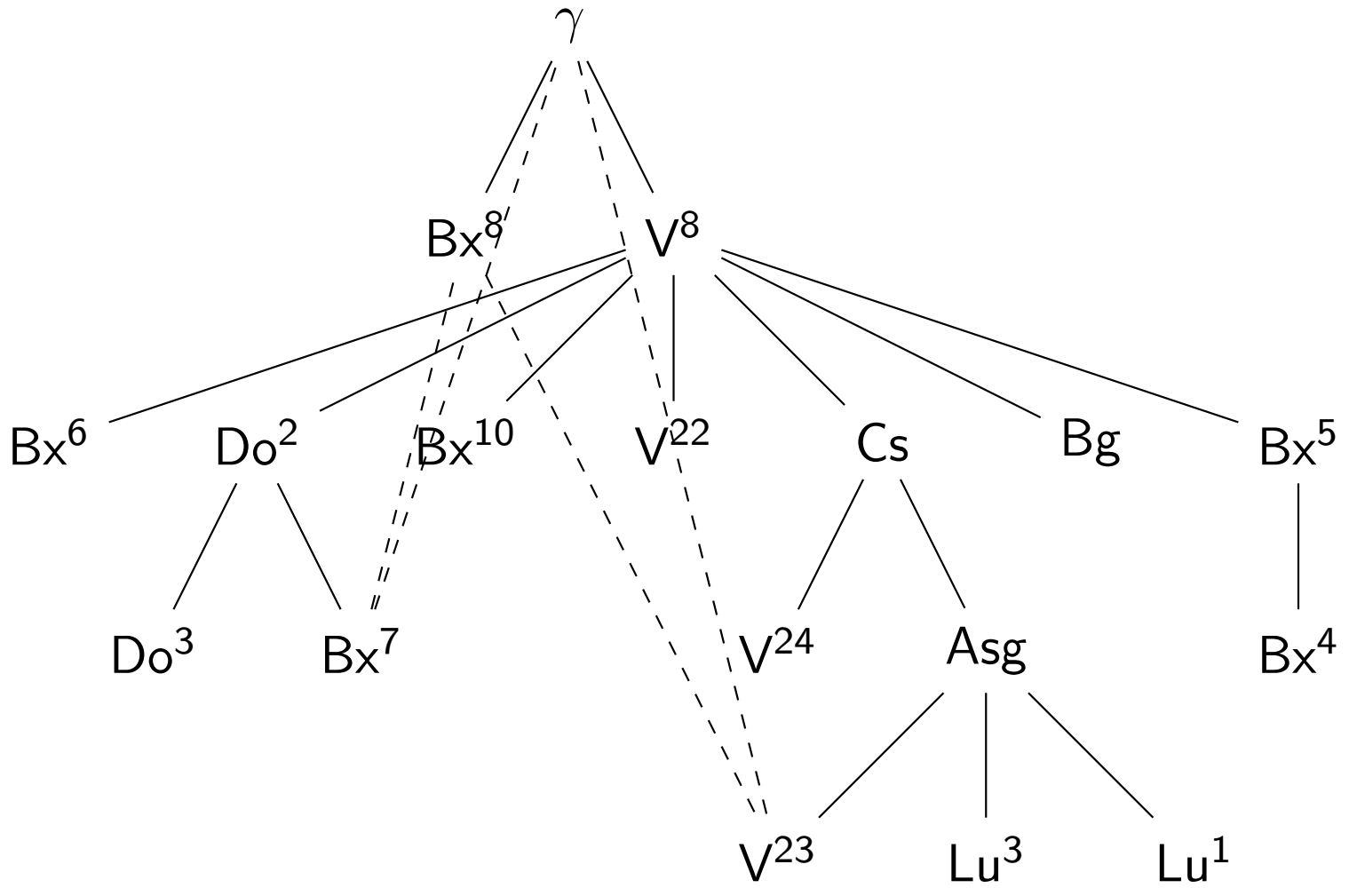
Collegamento di nodi arbitrari: caso reale

```
    child { node {Bg} }
    child { node {Bx$^{5}$}
      child { node {Bx$^{4}$} }
    }
  }
;
\draw[dashed,-] (B) -- (E);
\draw[dashed,-] (B) -- (g);
\draw[dashed,-] (E) -- (v23);
\draw[dashed,-] (g) -- (v23);
\end{tikzpicture}
```

Collegamento di nodi arbitrari: caso reale



Collegamento di nodi arbitrari: caso reale

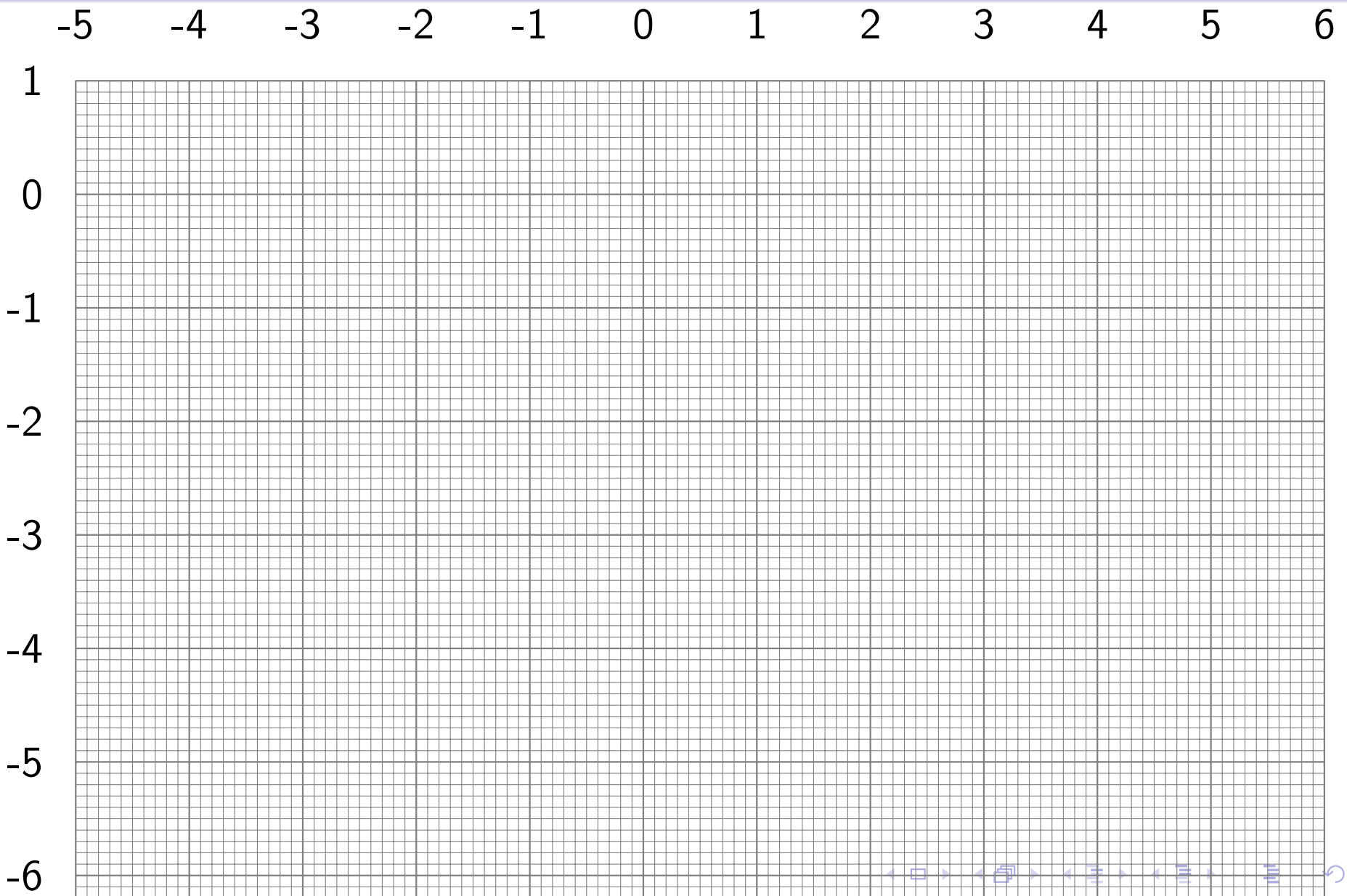


I nuovi archi sono sovrapposti ai nodi!

Una griglia di salvataggio

```
\begin{tikzpicture}
\draw[step=1cm,gray,thin] (-5,-7) grid (6,1);
\draw[step=1mm,gray,very thin] (-5,-7) grid (6,1);
\foreach \y in {-7,...,1}
  \node at (-5.5, \y)
    {\makebox[1pc][r]{\y}}; % numeri
      % imbandierati a destra
\foreach \x in {-5,...,6}
  \node at (\x, 1.5) {\x};
\end{tikzpicture}
```

Una griglia di salvataggio



Caso reale corretto

Ora possiamo posizionare i punti di controllo al meglio delle nostre esigenze:

```
\begin{tikzpicture}
[level 1/.style={sibling distance=1cm}]
\node (g) {$\gamma$}
  child { node (E) {Bx$^{8}$} }
  child { node {V$^{8}$}
    child { node {Bx$^{6}$} }
    child { node {Do$^{2}$}
      child { node {Do$^{3}$} }
      child { node (B) {Bx$^{7}$} }
    }
```

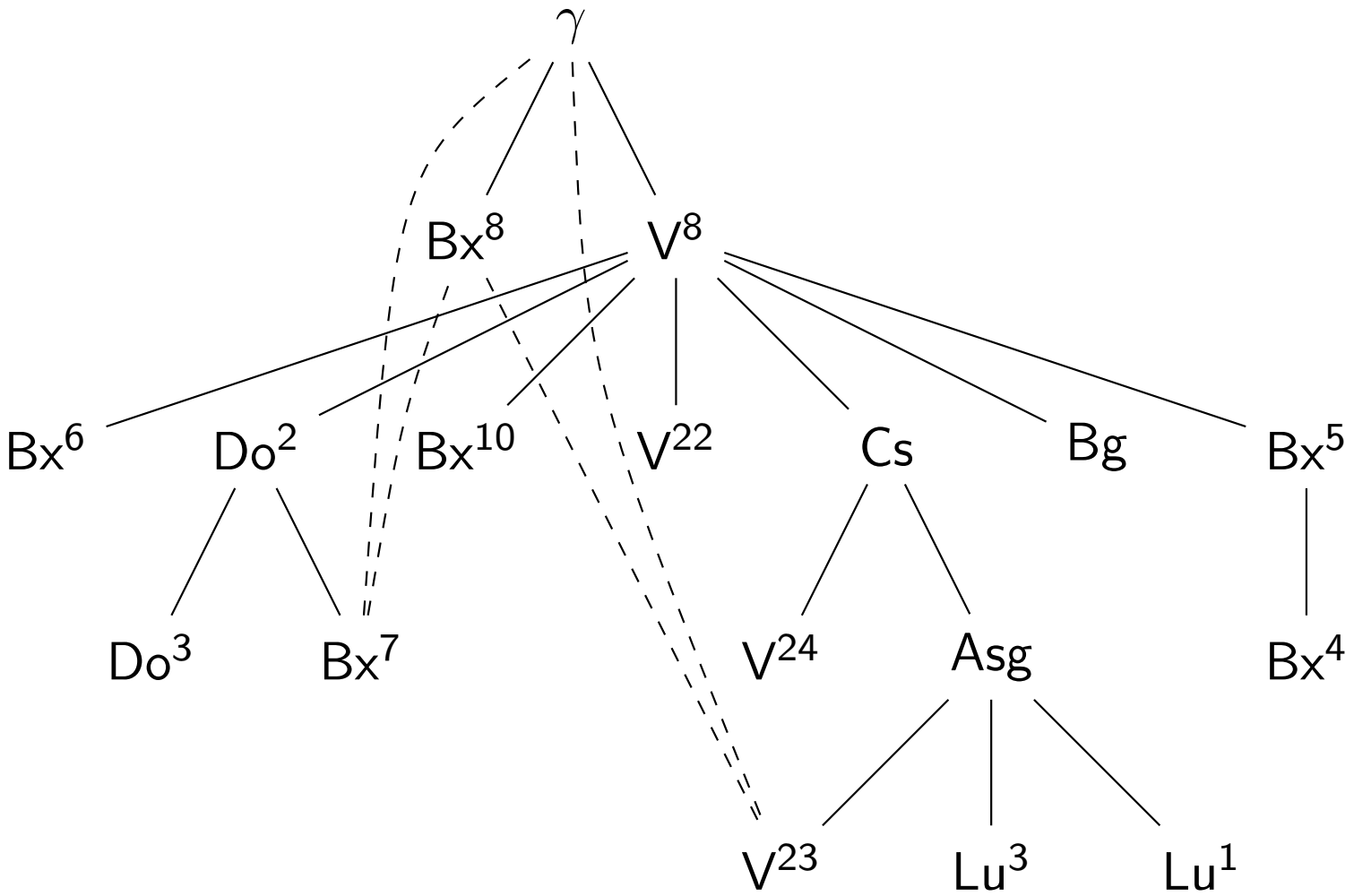
Caso reale corretto

```
child { node {Bx$^{10}$} }
child { node {V$^{22}$} }
child { node {Cs}
  child { node {V$^{24}$} }
  child { node {Asg}
    child { node (v23) {V$^{23}$} }
    child { node {Lu$^{3}$} }
    child { node {Lu$^{1}$} }
  }
}
```

Caso reale corretto

```
    child { node {Bg} }
    child { node {Bx$^{5}$}
      child { node {Bx$^{4}$} }
    }
  }
;
\draw[dashed,-] (B) ..
  controls (-1.2,-2.75) .. (E);
\draw[dashed,-] (B) ..
  controls (-1.25,-1) .. (g);
\draw[dashed,-] (E) -- (v23);
\draw[dashed,-] (g) ..
  controls (.1,-2.5) .. (v23);
\draw (a) -- (g);
\end{tikzpicture}
```

Caso reale corretto



Ti*k*Z vs `xytree`: primo round

Piccolo confronto tra grafi prodotti con `xytree` e con Ti*k*Z al meglio dei tre round

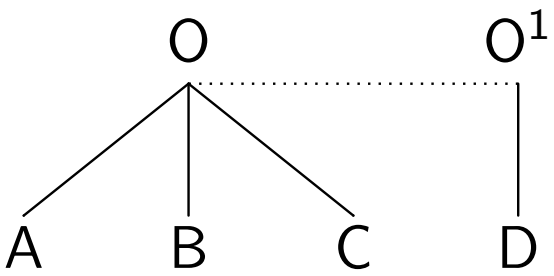
TikZ vs xytree: primo round

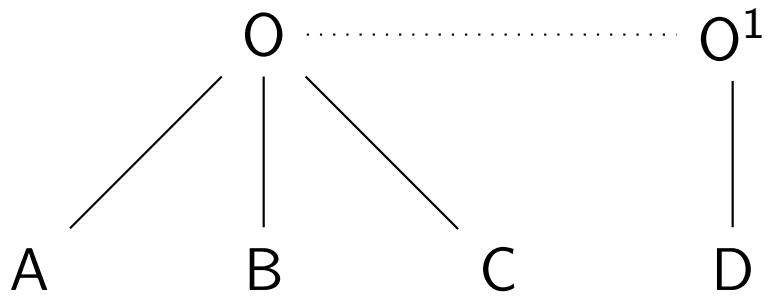
Piccolo confronto tra grafi prodotti con xytree e con TikZ al meglio dei tre round

```
\xytree{ & \xynode[-1,0,1]{O}
  \xyconnect[.](D,D){0,2} & &
\xynode[0]{O\ap{1}} \\
\xynode{A} & \xynode{B} &
\xynode{C} & \xynode{D}
}
```

```
\begin{tikzpicture}
\node(O){O}
  child{node {A}}
  child{node {B}}
  child{node {C}}
;
\node at (3,0) (O1){O$^1$}
  child{node {D}}
;
\draw[dotted] (O) -- (O1);
\end{tikzpicture}
```

TikZ vs xytree: primo round





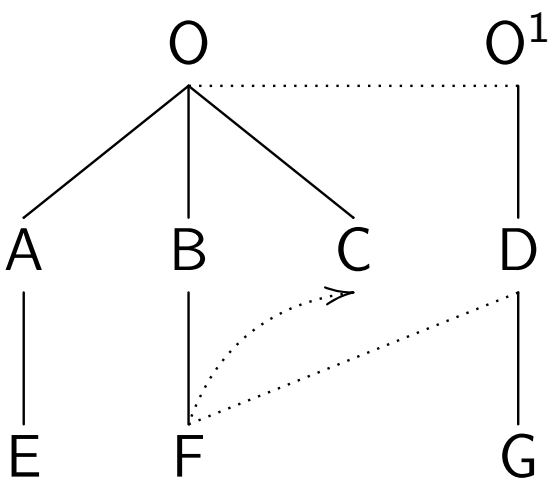
TikZ vs xytree: secondo round

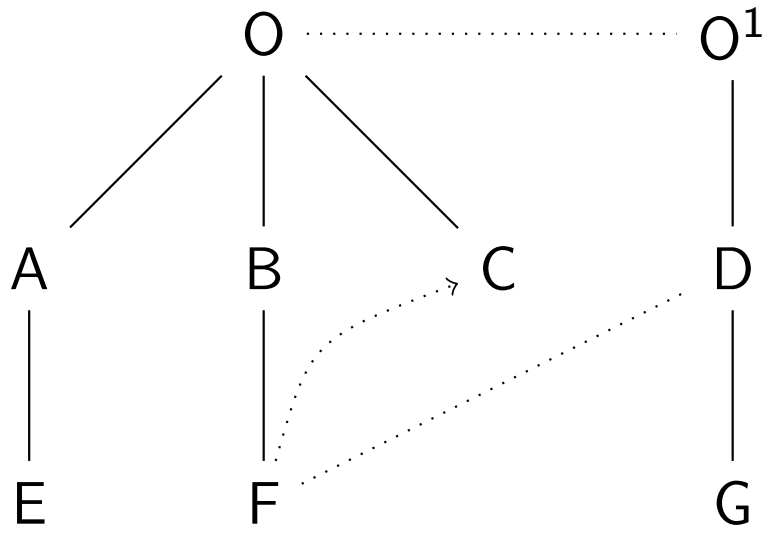
```
\xytree{ &
\xynode[-1,0,1]{O}
  \xyconnect[.](D,D){0,2} & &
\xynode[0]{O\ap{1}}\\
\xynode[0]{A} & \xynode[0]{B} &
\xynode{C} & \xynode[0]{D}
  \xyconnect[.](D,U){1,-2} \\
\xynode{E} & \xynode{F}
  \xyconnect[.>][^](U,D){-1,1} & &
\xynode{G}\\
}
```

TikZ vs xytree: secondo round

```
\begin{tikzpicture}
\node(O){O}
  child{node {A}
    child{node {E}}}
  child{node {B}
    child{node (F){F}}}
  child{node (C){C}}
;
\node at (3,0) (O1){O$^1$}
  child{node (D){D}
    child{node {G}}}
;
\draw[dotted] (O) -- (O1);
\draw[dotted] (F) -- (D);
\draw[dotted,->] (F) ..
 controls +(.3,1.05) .. (C);
\end{tikzpicture}
```

TikZ vs xytree: secondo round





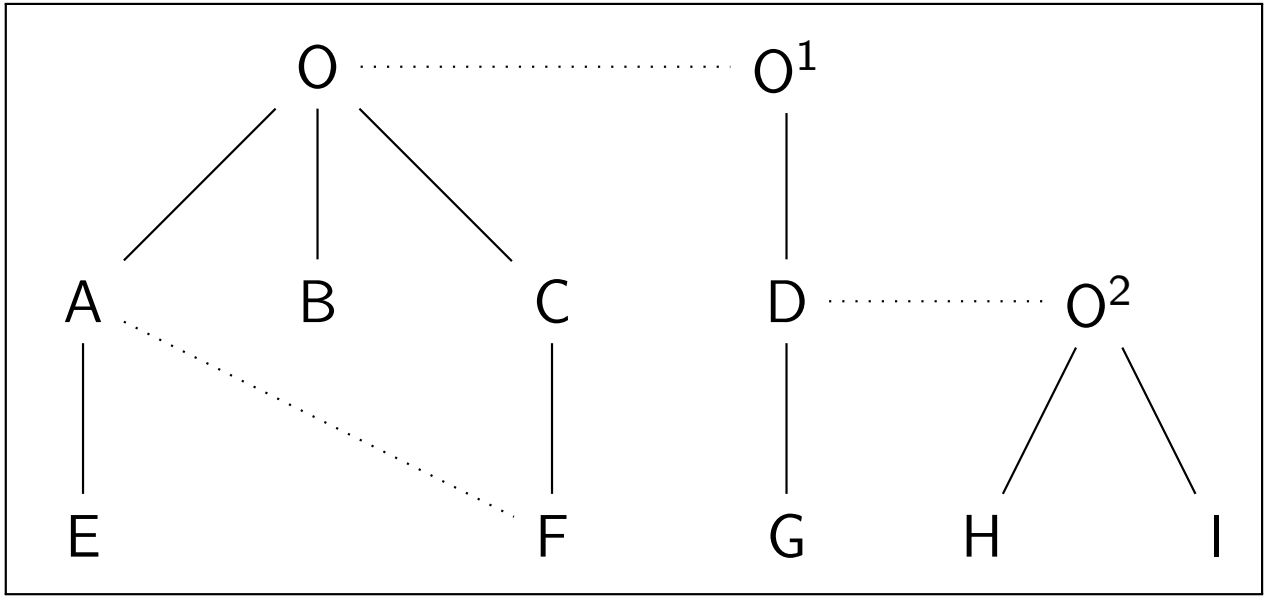
TikZ vs xytree: terzo round

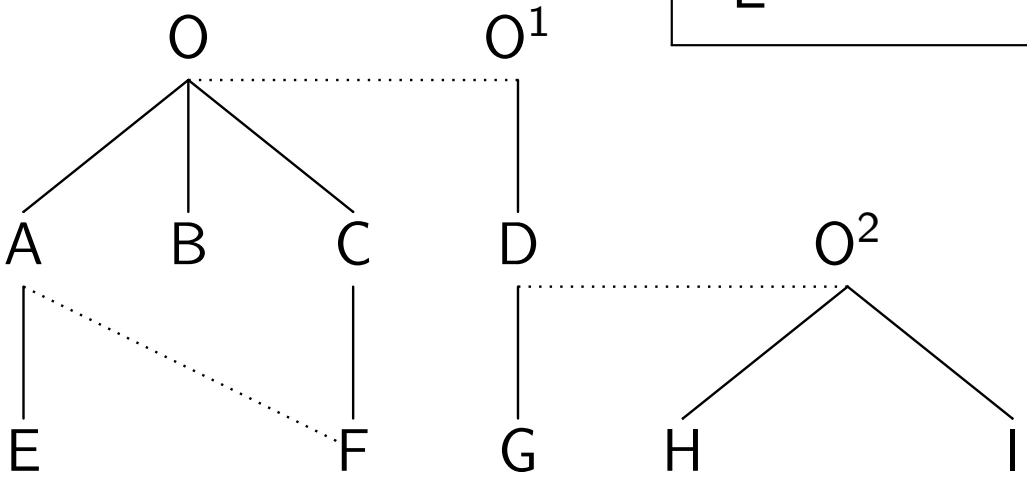
```
\xytree{ &
\xyconnect[.](D,D){0,2}
  \xynode[-1,0,1]{O} &%
&
\xynode[0]{O\ap{1}} & & & \\
\xyconnect[.](D,L){1,2} \xynode[0]{A} &
\xynode{B} & \xynode[0]{C} &
\xynode[0]{D} \xyconnect[.](D,D){0,2} & &
\xynode[-1,1]{O\ap{2}} &\\
\xynode{E}& &
\xynode{F} & \xynode{G} & \xynode{H} & &
\xynode{I} \\
}
```

TikZ vs xytree: terzo round

```
\begin{tikzpicture}
\node(O){O}
  child{node (A){A}
    child{node {E}}}
  child{node {B}}
  child{node {C}
    child{node (F){F}}}
;
\node at (3,0) (O1){O$^1$}
  child{node (D){D}
    child{node {G}}}
;
\node at (5,-1.5) (O2){O$^2$}
  child{node {H}}
  child{node {I}}
;
\draw[dotted] (O) -- (O1);
\draw[dotted] (A) -- (F);
\draw[dotted] (D) -- (O2);
\end{tikzpicture}
```

TikZ vs xytree: terzo round





TikZ vs `xytree`: il vincitore

Mentre `xytree` vuole una descrizione dell'*organizzazione fisica* del grafo, TikZ impone quella *logica*

TikZ vs xytree: il vincitore

Mentre xytree vuole una descrizione dell'*organizzazione fisica* del grafo, TikZ impone quella *logica*
TikZ non costringe a sapere in anticipo le posizioni dei singoli nodi dell'albero

TikZ vs `xytree`: il vincitore

Mentre `xytree` vuole una descrizione dell'*organizzazione fisica* del grafo, TikZ impone quella *logica*
TikZ non costringe a sapere in anticipo le posizioni dei singoli nodi dell'albero
L'aspetto dei grafi di `xytree` è raccolto e poco bilanciato: ricorda le tabelle con `\hline`

TikZ vs `xytree`: il vincitore

Mentre `xytree` vuole una descrizione dell'*organizzazione fisica* del grafo, TikZ impone quella *logica*
TikZ non costringe a sapere in anticipo le posizioni dei singoli nodi dell'albero
L'aspetto dei grafi di `xytree` è raccolto e poco bilanciato: ricorda le tabelle con `\hline`
TikZ offre un linguaggio piú lineare e consistente e un aspetto piú armonico