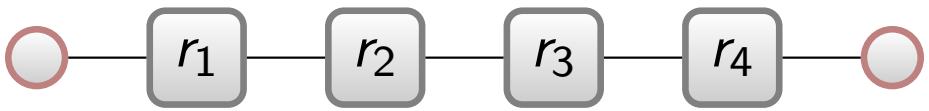
Fiabilidad

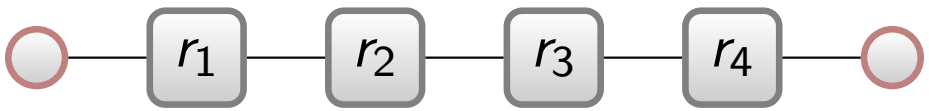
María Isabel Hartillo Hermoso
hartillo@us.es

Granada, 25 de Mayo
FQM-5849

Partimos de un sistema en serie:



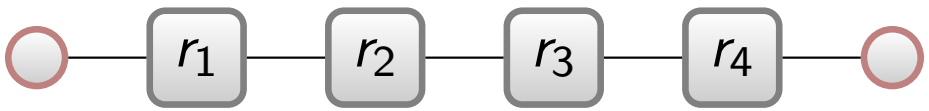
Partimos de un sistema en serie:



La fiabilidad del sistema es:

$$R = P(X_S = 1) =$$

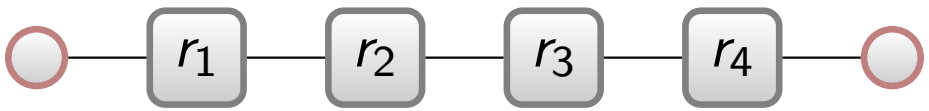
Partimos de un sistema en serie:



La fiabilidad del sistema es:

$$R = P(X_S = 1) = P(X_1 = 1, X_2 = 1, X_3 = 1, X_4 = 1) =$$

Partimos de un sistema en serie:

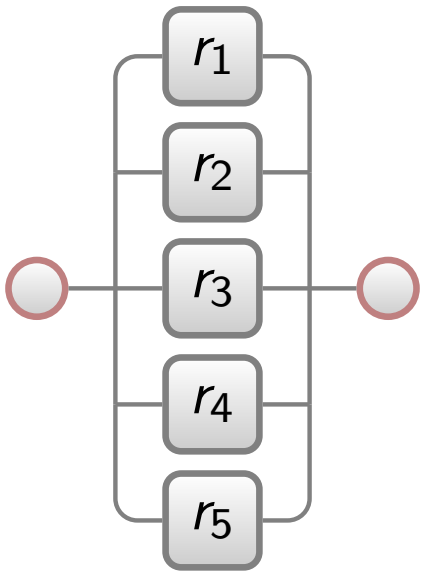


La fiabilidad del sistema es:

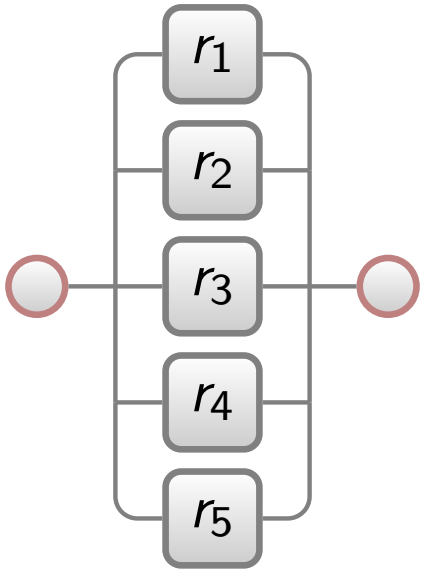
$$R = P(X_S = 1) = P(X_1 = 1, X_2 = 1, X_3 = 1, X_4 = 1) =$$

$$P(X_1 = 1) \cdot P(X_2 = 1) \cdot P(X_3 = 1) \cdot P(X_4 = 1) = r_1 \cdot r_2 \cdot r_3 \cdot r_4$$

Partimos de un sistema en paralelo:



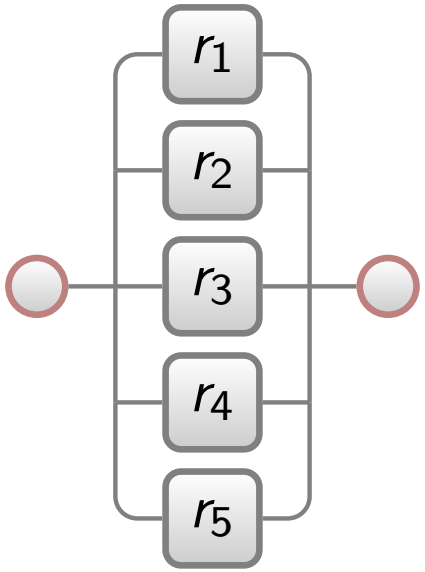
Partimos de un sistema en paralelo:



La fiabilidad del sistema es:

$$R = P(X_s = 1) = 1 - P(X_s = 0)$$

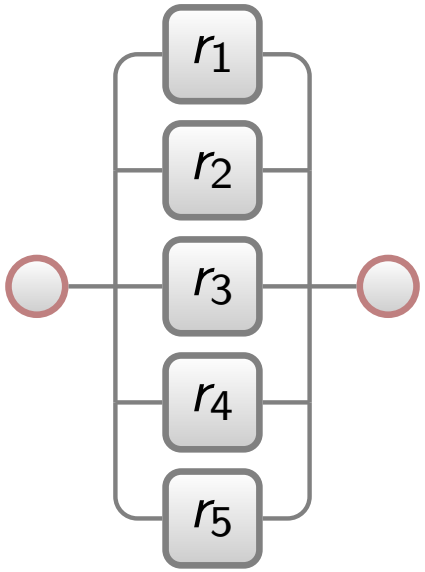
Partimos de un sistema en paralelo:



La fiabilidad del sistema es:

$$R = P(X_s = 1) = 1 - P(X_s = 0) = 1 - P(X_1 = 0, X_2 = 0, \ldots, X_5 = 0)$$

Partimos de un sistema en paralelo:

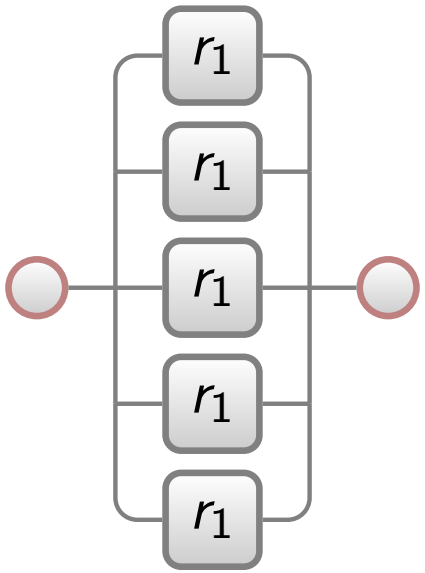


La fiabilidad del sistema es:

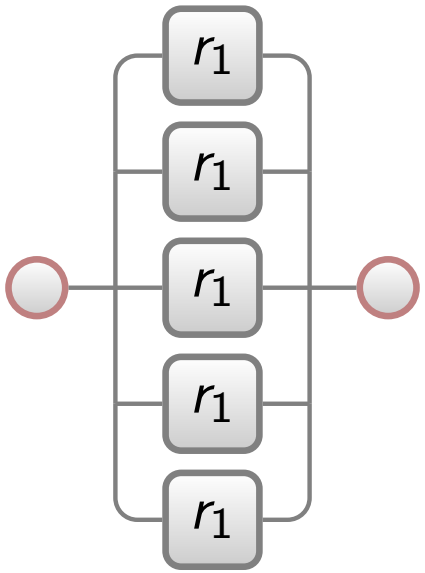
$$R = P(X_s = 1) = 1 - P(X_s = 0) = 1 - P(X_1 = 0, X_2 = 0, \ldots, X_5 = 0)$$

$$= 1 - P(X_1 = 0) \cdot P(X_2 = 0) \cdots P(X_5 = 0) = 1 - \prod_{i=1}^{5} (1 - r_i)$$

Partimos de un sistema en paralelo:



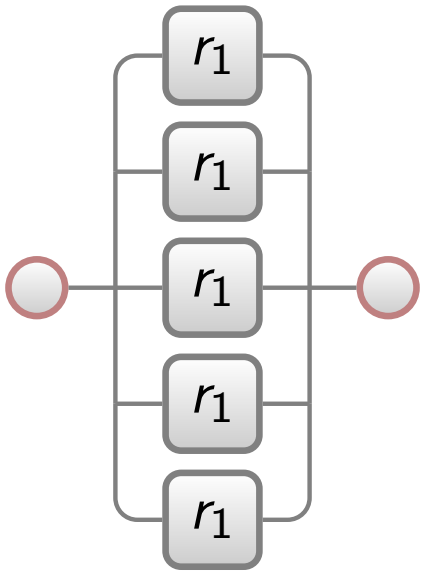
Partimos de un sistema en paralelo:



La fiabilidad del sistema es:

$$R = P(X_s = 1) = 1 - P(X_s = 0)$$

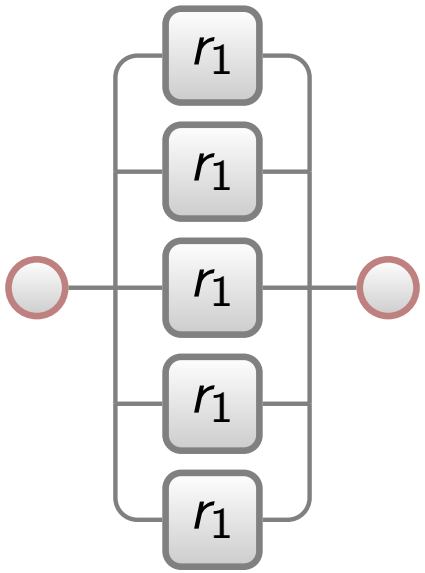
Partimos de un sistema en paralelo:



La fiabilidad del sistema es:

$$R = P(X_s = 1) = 1 - P(X_s = 0) = 1 - P(X_1 = 0, X_2 = 0, \ldots, X_5 = 0)$$

Partimos de un sistema en paralelo:

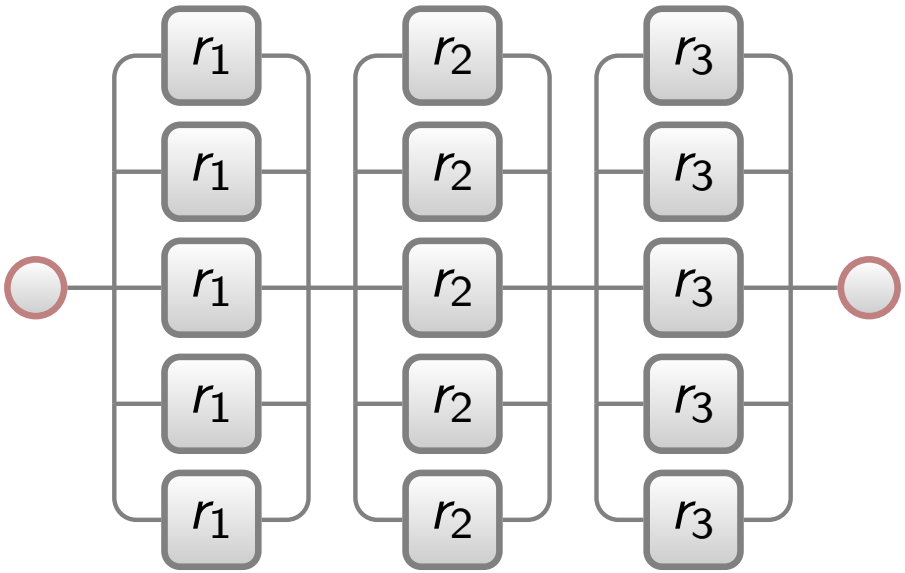


La fiabilidad del sistema es:

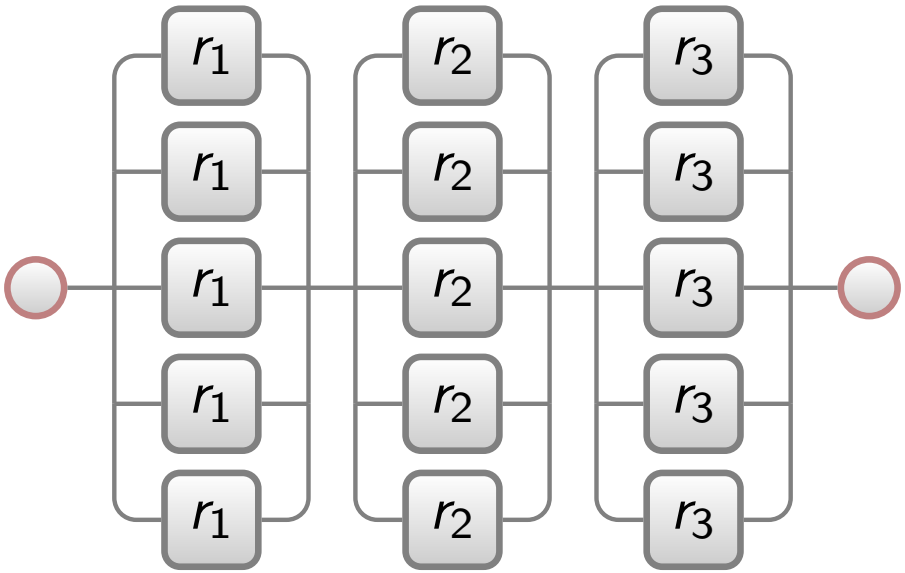
$$R = P(X_s = 1) = 1 - P(X_s = 0) = 1 - P(X_1 = 0, X_2 = 0, \ldots, X_5 = 0)$$

$$= 1 - P(X_1 = 0) \cdot P(X_2 = 0) \cdots P(X_5 = 0) = 1 - (1 - r_1)^5$$

Combinando, tenemos un sistema en serie paralelo:



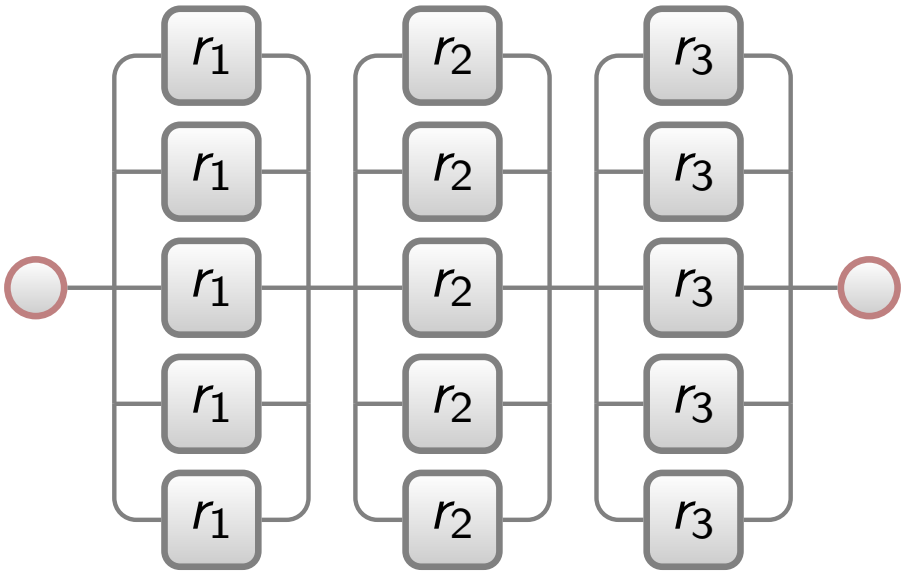
Combinando, tenemos un sistema en serie paralelo:



La fiabilidad del sistema es:

$$R = P(X_1 = 1) \cdot P(X_2 = 1) \cdot P(X_3 = 1)$$

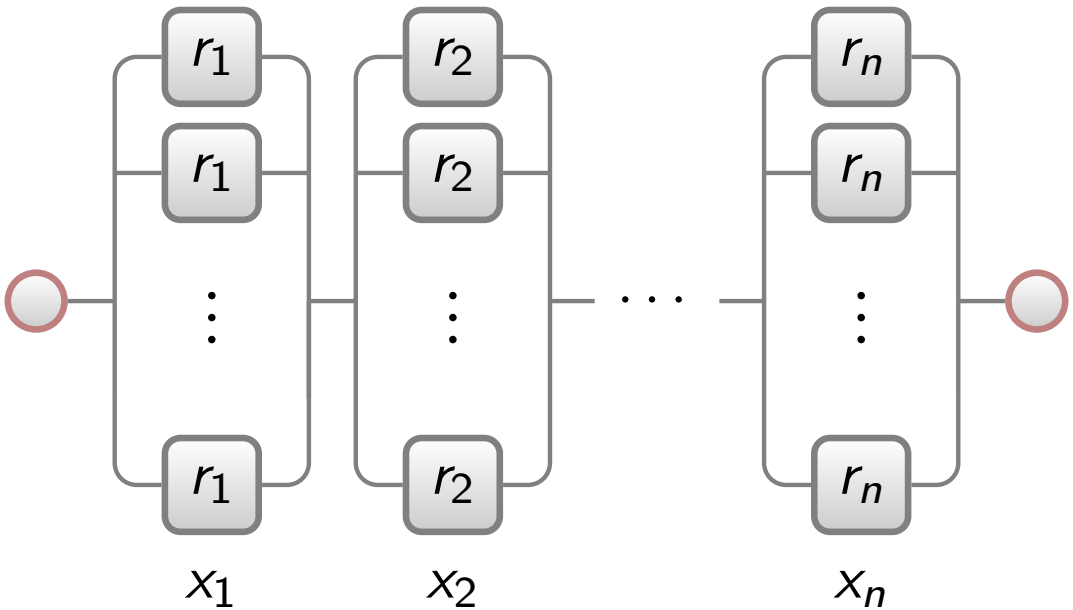
Combinando, tenemos un sistema en serie paralelo:



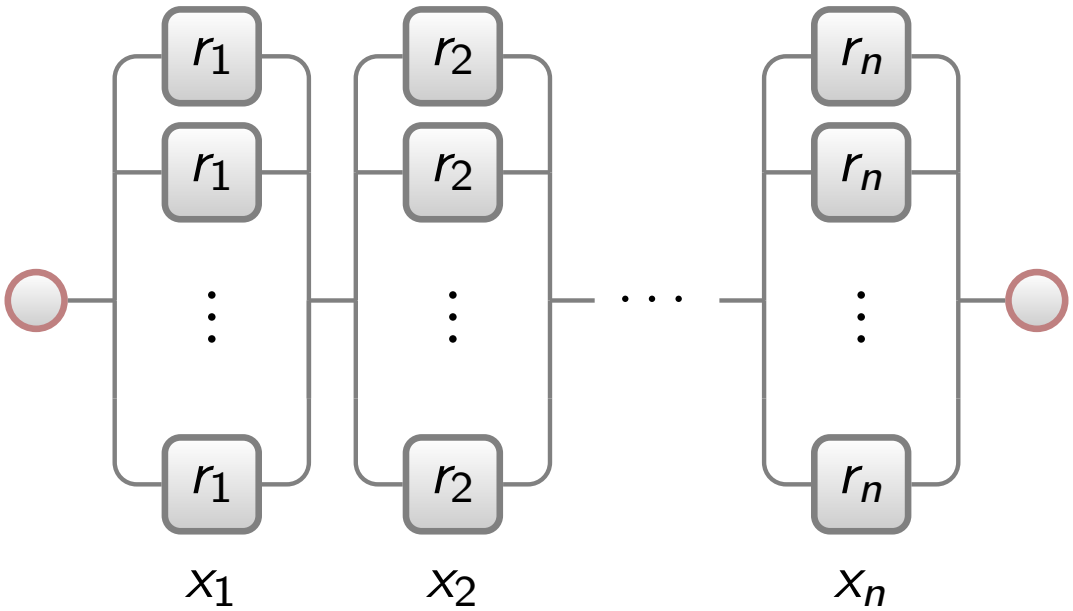
La fiabilidad del sistema es:

$$R = P(X_1 = 1) \cdot P(X_2 = 1) \cdot P(X_3 = 1) = \prod_{i=1}^{3} (1 - (1 - r_i)^5)$$

Consideramos un sistema en serie, con n subsistemas en paralelo:

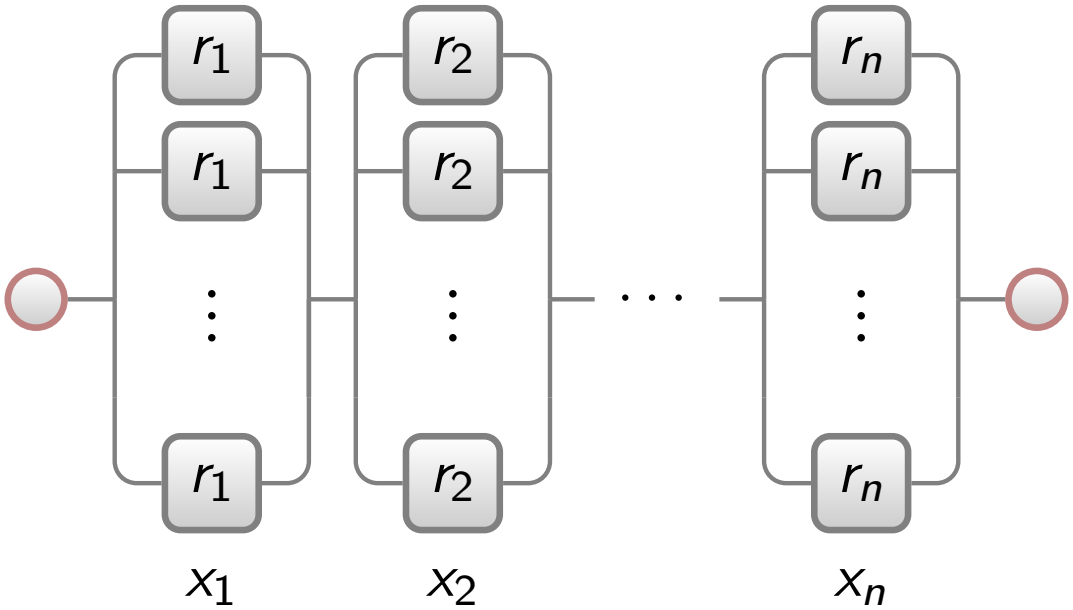


Consideramos un sistema en serie, con n subsistemas en paralelo:



$$R(x) = \prod_{i=1}^{n} (1 - (1 - r_i)^{x_i})$$

Consideramos un sistema en serie, con n subsistemas en paralelo:

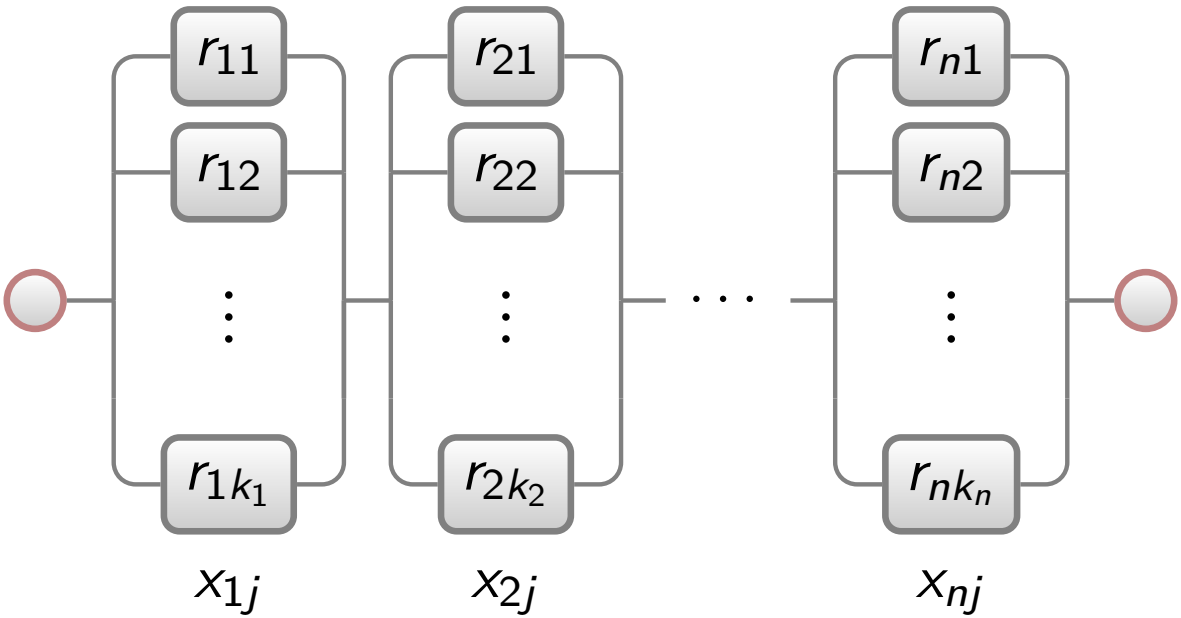


$$(P1) \qquad \text{mín} \sum_i c_i x_i$$

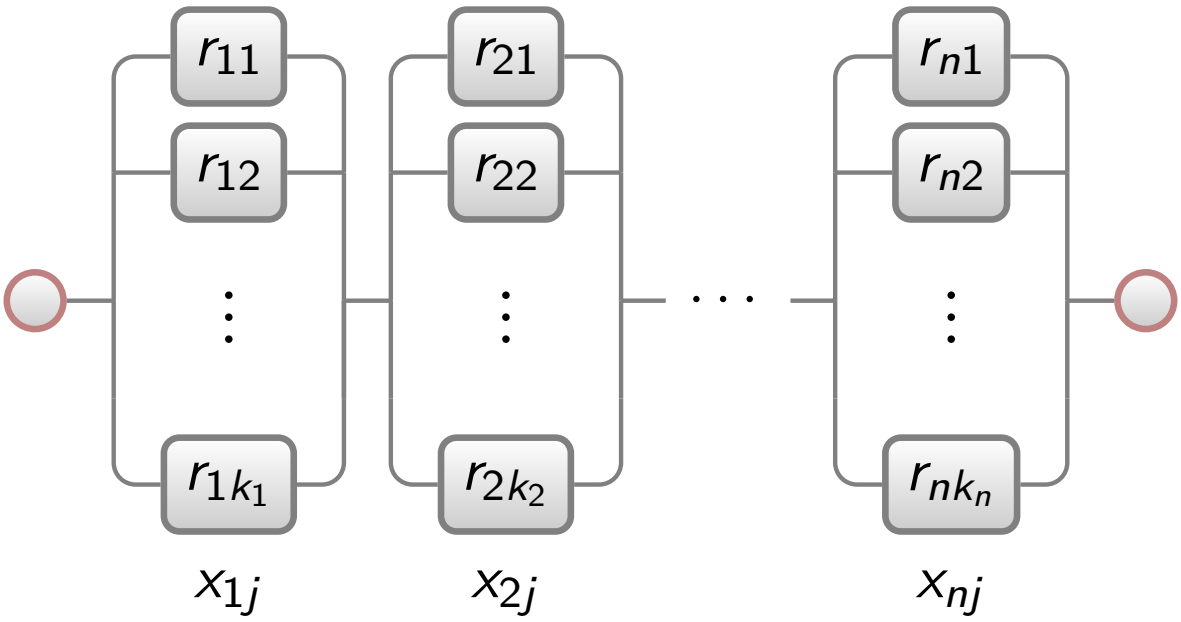
$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - (1 - r_i)^{x_i}) \geq R_0,$$

$$1 \leq x_i \leq u_i$$

Si consideramos diferentes componentes en cada subsistema:

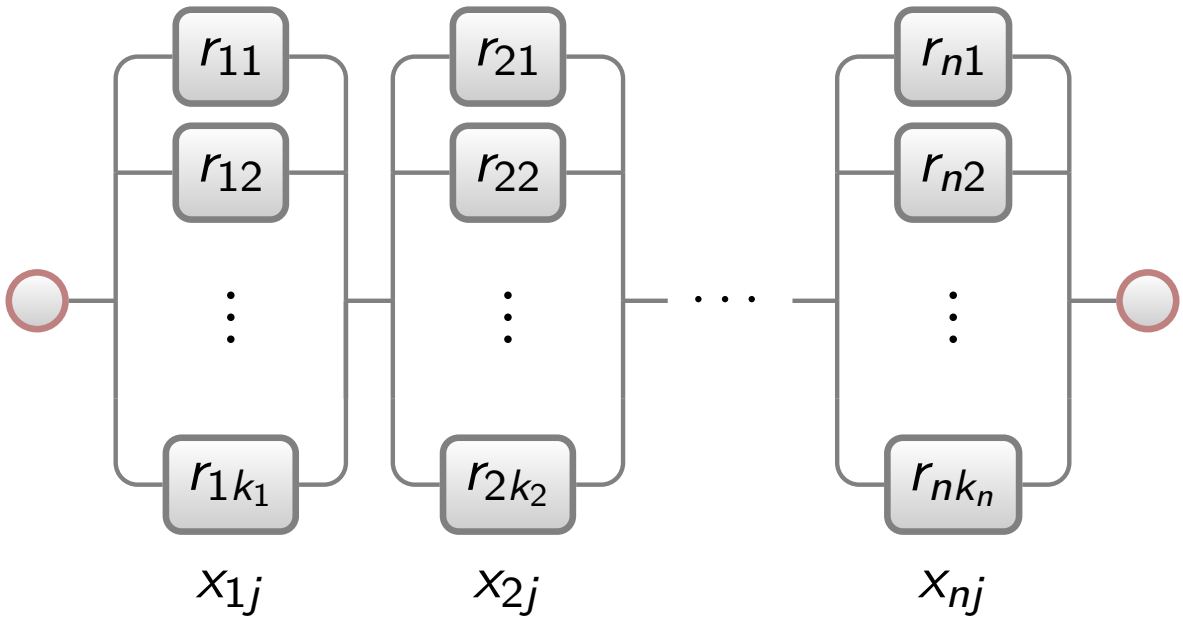


Si consideramos diferentes componentes en cada subsistema:



$$R(x) = \prod_{i=1}^{n}(1 - \prod_{j=1}^{k_j}(1 - r_{ij})^{x_{ij}})$$

Si consideramos diferentes componentes en cada subsistema:

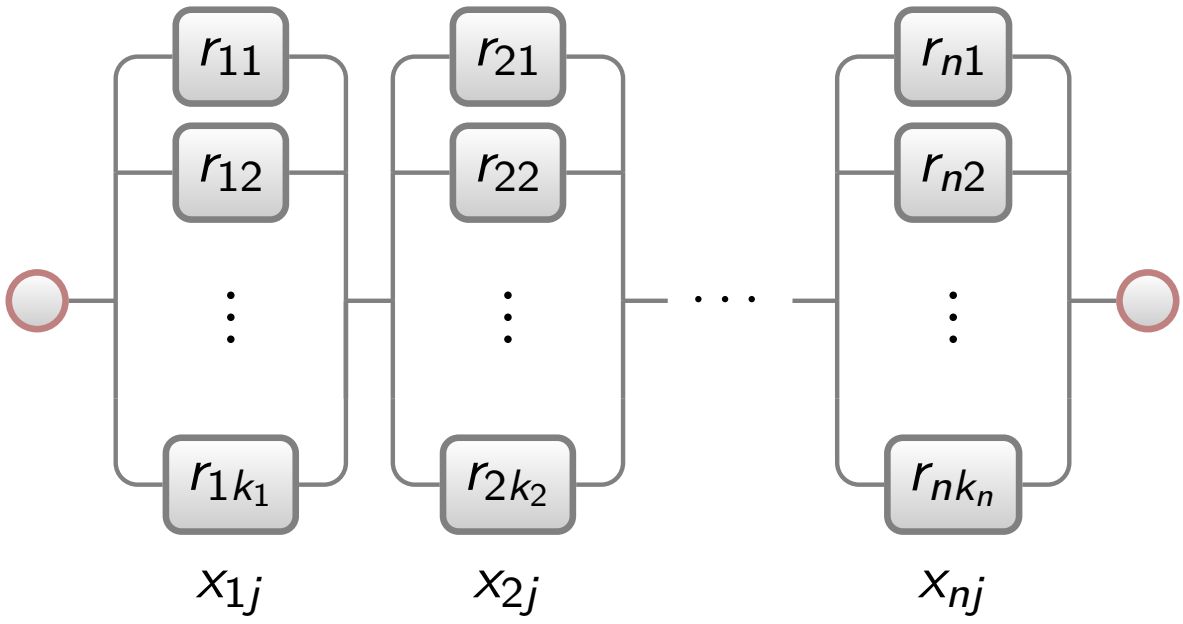


$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_j} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Si consideramos diferentes componentes en cada subsistema:



$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_j} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

Optimizando la fiabilidad

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.

Optimización

C.S. Sung, Y.K. Cho.
Reliability optimization of a series system with multiple-choice and budget constraint.
European Journal of Operational Research, 75(1): 217–232, 2000.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.

Aproximación

F. Ahmadizar, H. Soltanpanah.
Reliability optimization of a series system with multiple-choice and budget constraints using an efficient ant colony approach.
Expert Systems with Applications, 38(4): 3640–3646, 2011.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.
- Podemos minimizar el coste, fijada una fiabilidad mínima.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.
- Podemos minimizar el coste, fijada una fiabilidad mínima.

Optimización

M. Djerdjour, K. Rekab. A branch and bound algorithm for designing reliable systems at a minimum cost. *Applied Mathematics and Computation*, 118(2): 247–259, 2001.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.
- Podemos minimizar el coste, fijada una fiabilidad mínima.

Optimización

N. Ruan, X. Sun.
An exact algorithm for cost minimization in series reliability systems with multiple component choices.
Applied Mathematics and Computation, 181(1): 732–741, 2006.

Optimizando la fiabilidad

- Cantidad óptima de componentes redundantes, en cada subsistema.
- Maximizar la fiabilidad, sujeto a un presupuesto.
- Podemos minimizar el coste, fijada una fiabilidad mínima.

Aproximación

J.E. Ramirez-Marquez, D.W. Coit.
A heuristic for solving the redundancy allocation problem for multi-state series-parallel systems.
Reliability Engineering & System Safety, 83(3): 341–349, 2004.

$$(P1) \qquad \text{mín} \sum_i c_i x_i$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - (1 - r_i)^{x_i}) \geq R_0,$$

$$1 \leq l_i \leq x_i \leq u_i$$

$$(P1) \qquad \text{mín} \sum_i c_i x_i$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - (1 - r_i)^{x_i}) \geq R_0,$$

$$1 \leq l_i \leq x_i \leq u_i$$

$$y_i = u_i - x_i$$

$$y_i = u_i - x_i$$

$$(MP1) \qquad \max \sum_i c_i y_i$$

$$\text{s.t.} \quad \sum_{i=1}^{n} -\log(1-(1-r_i)^{u_i - y_i}) \leq -\log(R_0),$$

$$0 \leq y_i \leq u_i - l_i$$

$$y_i = u_i - x_i$$

$$(MP1) \qquad \max \sum_i c_i y_i$$

$$\text{s.t.} \quad \sum_{i=1}^{n} -\log(1-(1-r_i)^{u_i - y_i}) \leq -\log(R_0),$$

$$0 \leq y_i \leq u_i - l_i$$

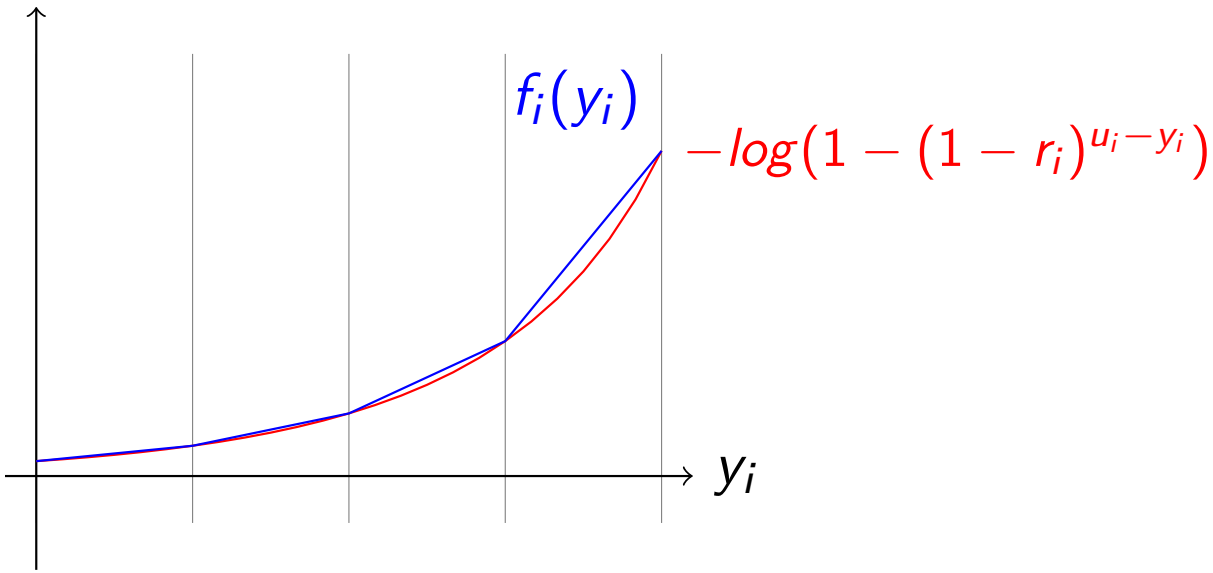
Como $R(y)$ es separable, podemos construir una función $f_i(y_i)$ lineal a trozos con uniones en los valores enteros

Como $R(y)$ es separable, podemos construir una función $f_i(y_i)$ lineal a trozos con uniones en los valores enteros

$$\begin{array}{l|c} f_i(0) & -\log(1-(1-r_i)^{u_i}) \\ f_i(1) & -\log(1-(1-r_i)^{u_i-1}) \\ \vdots & \vdots \\ f_i(u_i-l_i) & -\log(1-(1-r_i)^{l_i}) \end{array}$$

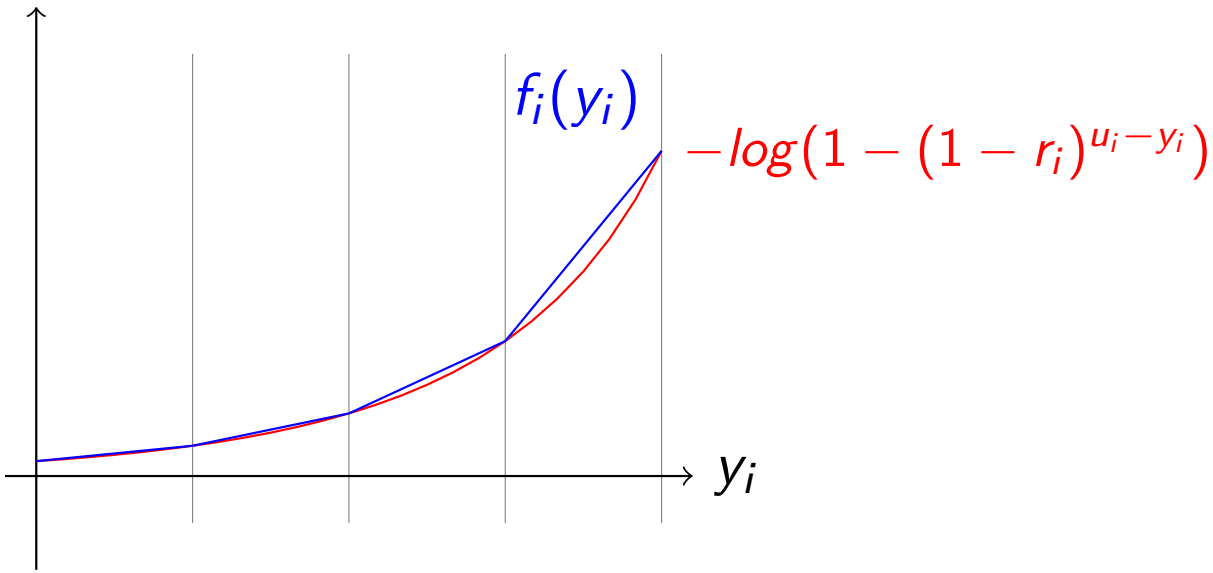
Como $R(y)$ es separable, podemos construir una función $f_i(y_i)$ lineal a trozos con uniones en los valores enteros

$$\begin{array}{l|l} f_i(0) & -\log(1-(1-r_i)^{u_i}) \\ f_i(1) & -\log(1-(1-r_i)^{u_i-1}) \\ \vdots & \vdots \\ f_i(u_i-l_i) & -\log(1-(1-r_i)^{l_i}) \end{array}$$



Como $R(y)$ es separable, podemos construir una función $f_i(y_i)$ lineal a trozos con uniones en los valores enteros

$$\begin{array}{l|l} f_i(0) & -\log(1-(1-r_i)^{u_i}) \\ f_i(1) & -\log(1-(1-r_i)^{u_i-1}) \\ \vdots & \vdots \\ f_i(u_i-l_i) & -\log(1-(1-r_i)^{l_i}) \end{array}$$



$f_i(y_i)$ es función convexa no decreciente.

Si denominamos $f_{ik} = f_i(k)$, (MP1) es equivalente a:

$$(MP1') \qquad \text{máx} \sum_i \sum_k c_i w_{ik}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{k=1}^{u_i - l_i} (f_{ik} - f_{ik-1}) w_{ik} \leq -\log(R_0),$$

$$w_{ik} = 0,\ 1$$

$$w_{ik} > 0 \Rightarrow w_{ij} = 1 \quad \forall j < k$$

Algoritmo greedy

$y^0 = (0, \ldots, 0)$ $c^0 = \left(\frac{c_1}{f_1(1)-f_1(0)}, \ldots, \frac{c_n}{f_n(1)-f_n(0)} \right)$.

repeat

$i^* = \text{máx}\{c_1^0, \ldots c_n^0\}$

$y_{i^*}^0 = y_{i^*}^0 + \text{mín} \left\{ 1, \frac{-\log(R_0)+\log(R(y^0))}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})} \right\}$

$c_{i*}^0 = \frac{{c_i}^*}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}$

until $y_i^0 = u_i - l_i$ or $R(y^0) = R_0$

Algoritmo greedy

$y^0 = (0, \ldots, 0)$ $c^0 = \left(\frac{c_1}{f_1(1)-f_1(0)}, \ldots, \frac{c_n}{f_n(1)-f_n(0)} \right)$.

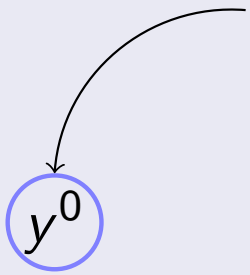
repeat

$i^* = \text{máx}\{c_1^0, \ldots c_n^0\}$

$y_{i^*}^0 = y_{i^*}^0 + \text{mín} \left\{ 1, \frac{-\log(R_0)+\log(R(y^0))}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})} \right\}$

$c_{i*}^0 = \frac{{c_i}^*}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}$

until $y_i^0 = u_i - l_i$ or $R(y^0) = R_0$



Algoritmo greedy

$y^0 = (0, \ldots, 0)$ $c^0 = \left(\frac{c_1}{f_1(1)-f_1(0)}, \ldots, \frac{c_n}{f_n(1)-f_n(0)} \right)$.

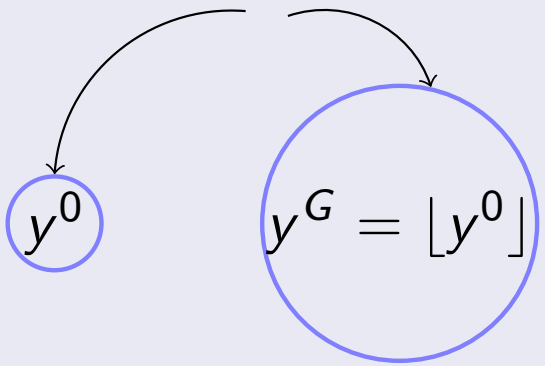
repeat

$i^* = \text{máx}\{c_1^0, \ldots c_n^0\}$

$y_{i^*}^0 = y_{i^*}^0 + \text{mín} \left\{ 1, \frac{-\log(R_0)+\log(R(y^0))}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})} \right\}$

$c_{i^*}^0 = \frac{{c_i}^*}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}$

until $y_i^0 = u_i - l_i$ or $R(y^0) = R_0$



Algoritmo greedy

$y^0 = (0, \ldots, 0)$ $c^0 = \left(\frac{c_1}{f_1(1)-f_1(0)}, \ldots, \frac{c_n}{f_n(1)-f_n(0)} \right)$.

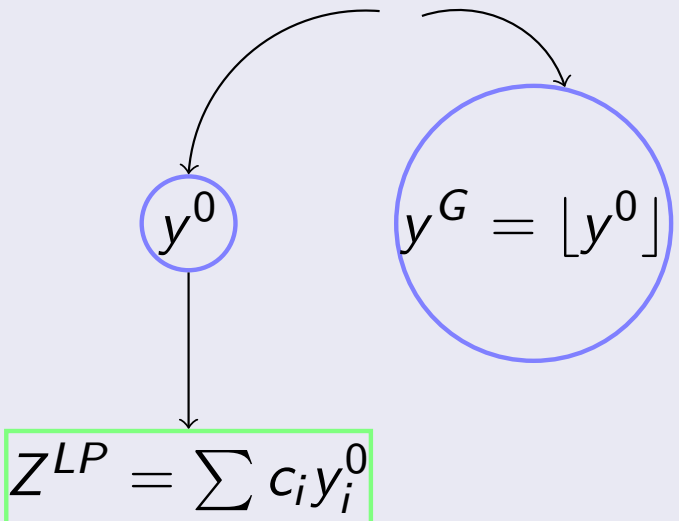
repeat

$i^* = \text{máx}\{c_1^0, \ldots c_n^0\}$

$y_{i^*}^0 = y_{i^*}^0 + \text{mín} \left\{ 1, \frac{-\log(R_0)+\log(R(y^0))}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})} \right\}$

$c_{i^*}^0 = \frac{c_{i^*}}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}$

until $y_i^0 = u_i - l_i$ or $R(y^0) = R_0$



Algoritmo greedy

$y^0 = (0, \ldots, 0)$ $c^0 = \left(\frac{c_1}{f_1(1)-f_1(0)}, \ldots, \frac{c_n}{f_n(1)-f_n(0)} \right)$.

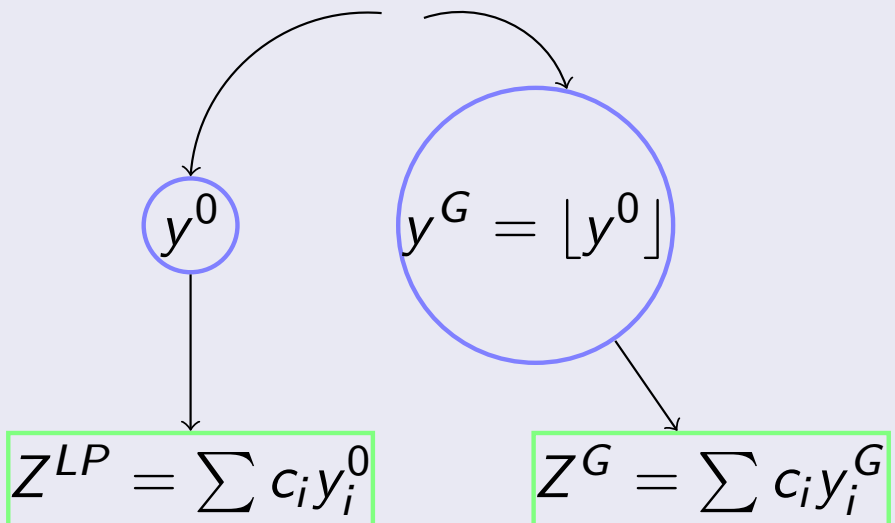
repeat

$i^* = \text{máx}\{c_1^0, \ldots c_n^0\}$

$y_{i^*}^0 = y_{i^*}^0 + \text{mín}\left\{1, \frac{-\log(R_0)+\log(R(y^0))}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}\right\}$

$c_{i^*}^0 = \frac{c_{i^*}}{f_{i^*}(y_{i^*}+1)-f_{i^*}(y_{i^*})}$

until $y_i^0 = u_i - l_i$ or $R(y^0) = R_0$



$$(P2) \qquad \min \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

Relajamos el problema a uno lineal:

$$(RP2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq n \log(1 - R_0^{1/n}),$$

$$\sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Relajación lagrangiana D^1

$$d^1(\lambda) = \quad \text{mín} \sum_{i,j} c_{ij}x_{ij} + \lambda\left(\sum_{ij} a_{ij}x_{ij} - b_0\right)$$

s.t.

$$\sum_{j=1}^{k_i} a_{ij}x_{ij} \leq b_i$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Relajación lagrangiana D^1

$$d^1(\lambda) = \quad \text{mín} \sum_{i,j} c_{ij}x_{ij} + \lambda\left(\sum_{ij} a_{ij}x_{ij} - b_0\right)$$

s.t.

$$\sum_{j=1}^{k_i} a_{ij}x_{ij} \leq b_i$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Con problema dual

$$(D^1) \max_{\lambda \geq 0} d^1(\lambda)$$

Relajación lagrangiana D^1

$$d^1(\lambda) = \begin{array}{r} \text{mín} \sum_{i,j} c_{ij}x_{ij} + \\ \lambda\left(\sum_{ij} a_{ij}x_{ij} - b_0\right) \end{array}$$

s.t.

$$\sum_{j=1}^{k_i} a_{ij}x_{ij} \leq b_i$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Con problema dual

$$(D^1) \max_{\lambda \geq 0} d^1(\lambda)$$

Relajación lagrangiana D^2

$$d^2(\lambda) = \begin{array}{r} \text{mín} \sum_{i,j} c_{ij}x_{ij} + \\ \lambda_0(\sum_{ij} a_{ij}x_{ij} - b_0)) \\ + \sum_i \lambda_i(\sum_j a_{ij}x_{ij} - b_i) \end{array}$$

s.t.

$$0 \leq x_{i,j} \leq u_{i,j}$$

Relajación lagrangiana D^1

$$d^1(\lambda) = \min \sum_{i,j} c_{ij}x_{ij} + \lambda\left(\sum_{ij} a_{ij}x_{ij} - b_0\right)$$

s.t.

$$\sum_{j=1}^{k_i} a_{ij}x_{ij} \leq b_i$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Con problema dual

$$(D^1) \max_{\lambda \geq 0} d^1(\lambda)$$

Relajación lagrangiana D^2

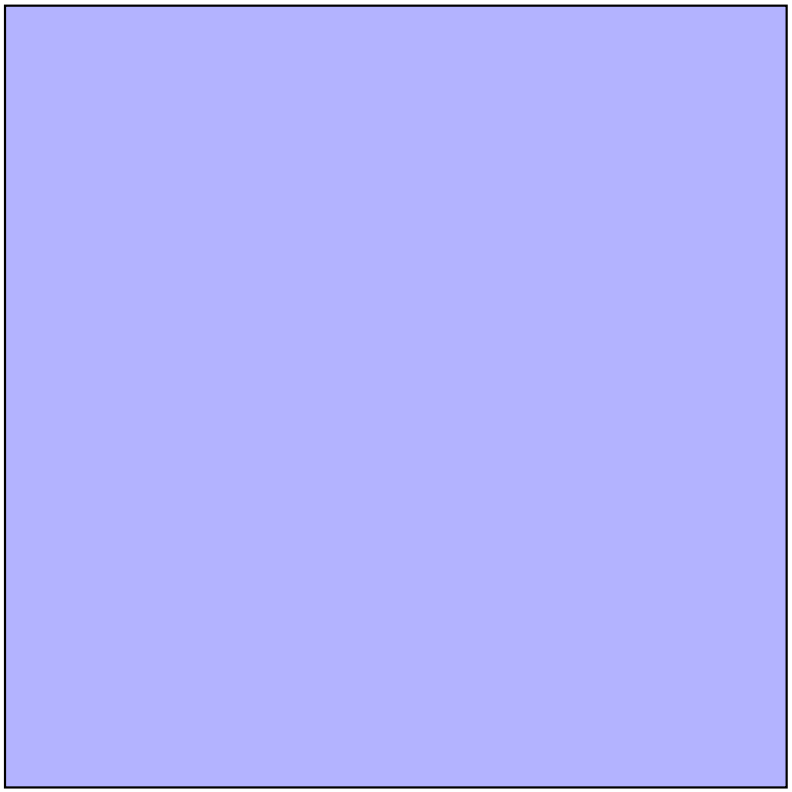
$$d^2(\lambda) = \min \sum_{i,j} c_{ij}x_{ij} + \lambda_0\left(\sum_{ij} a_{ij}x_{ij} - b_0\right) + \sum_i \lambda_i\left(\sum_j a_{ij}x_{ij} - b_i\right)$$

s.t.

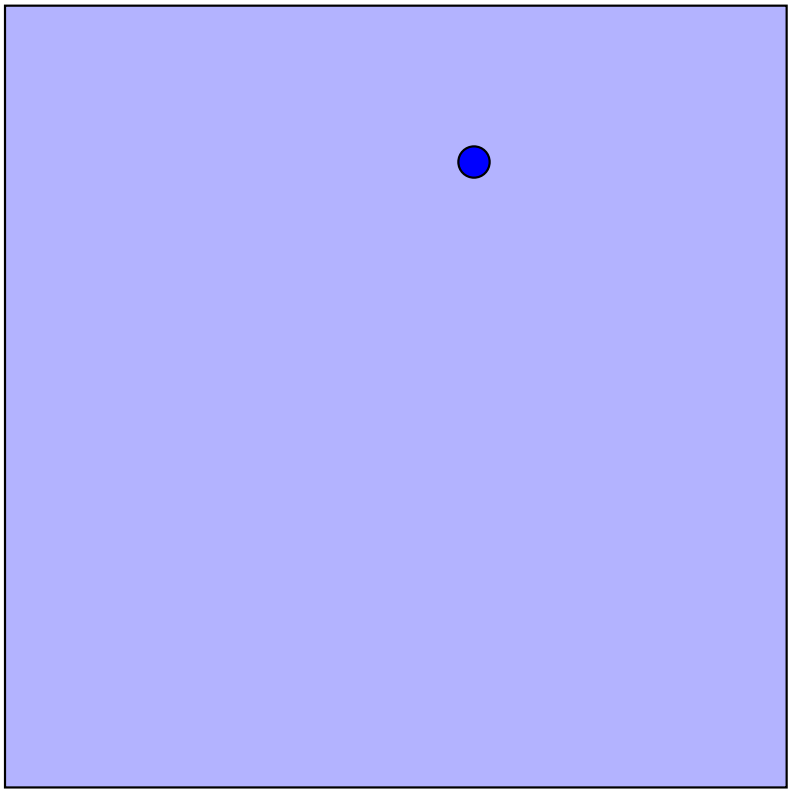
$$0 \leq x_{i,j} \leq u_{i,j}$$

Con problema dual

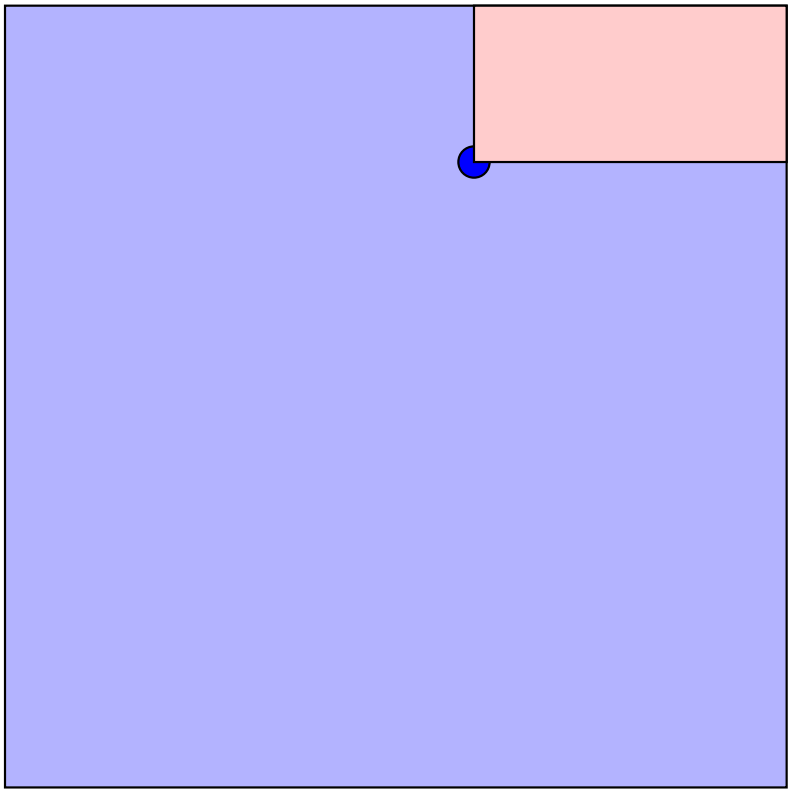
$$(D^2) \max_{\lambda \in \mathbb{R}_+^{n+1}} d^2(\lambda)$$



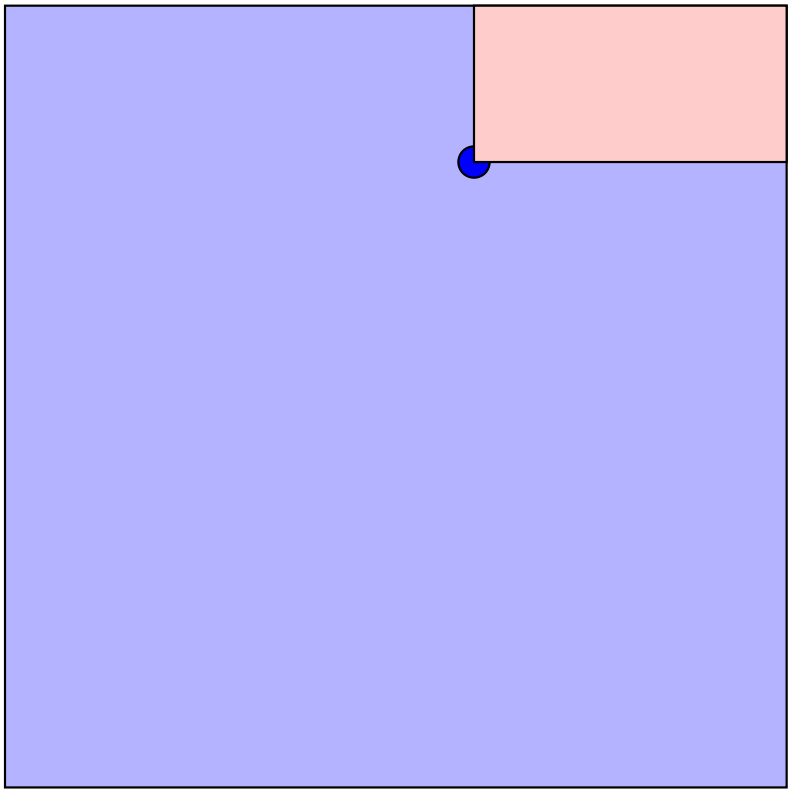
Partición 1



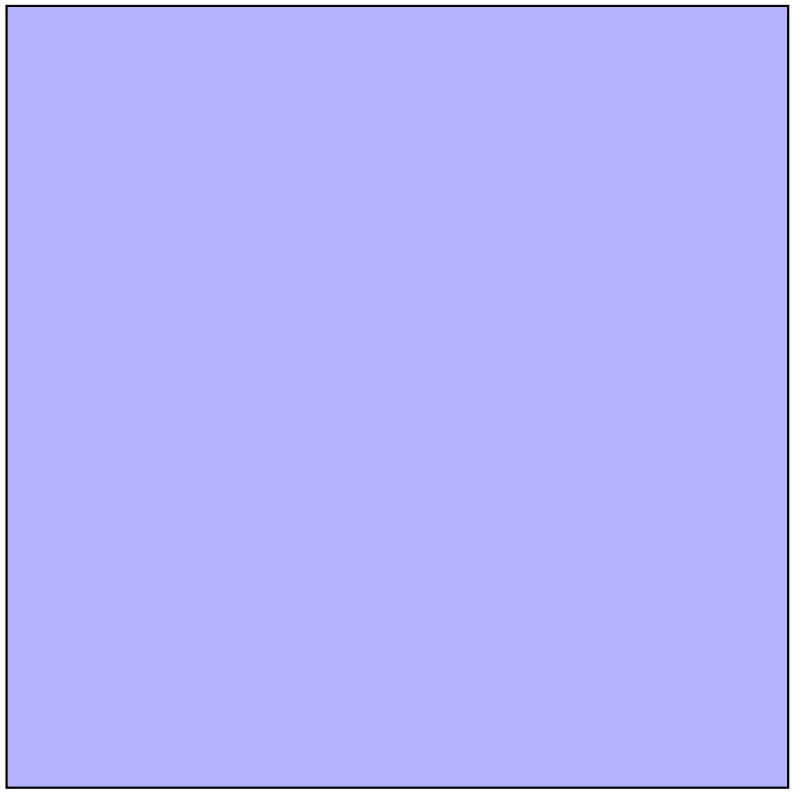
Partición 1



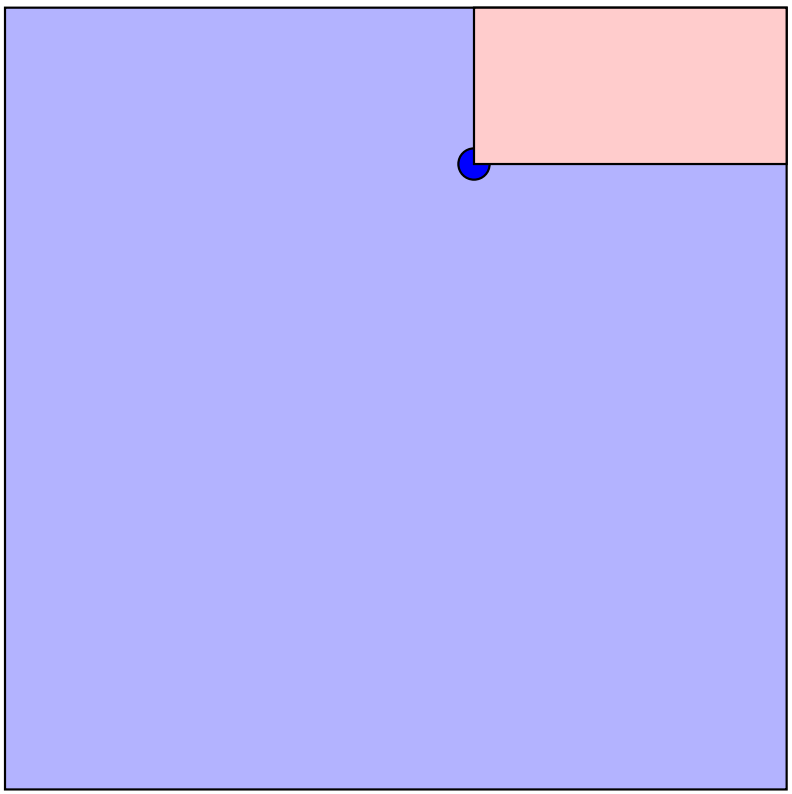
Partición 1



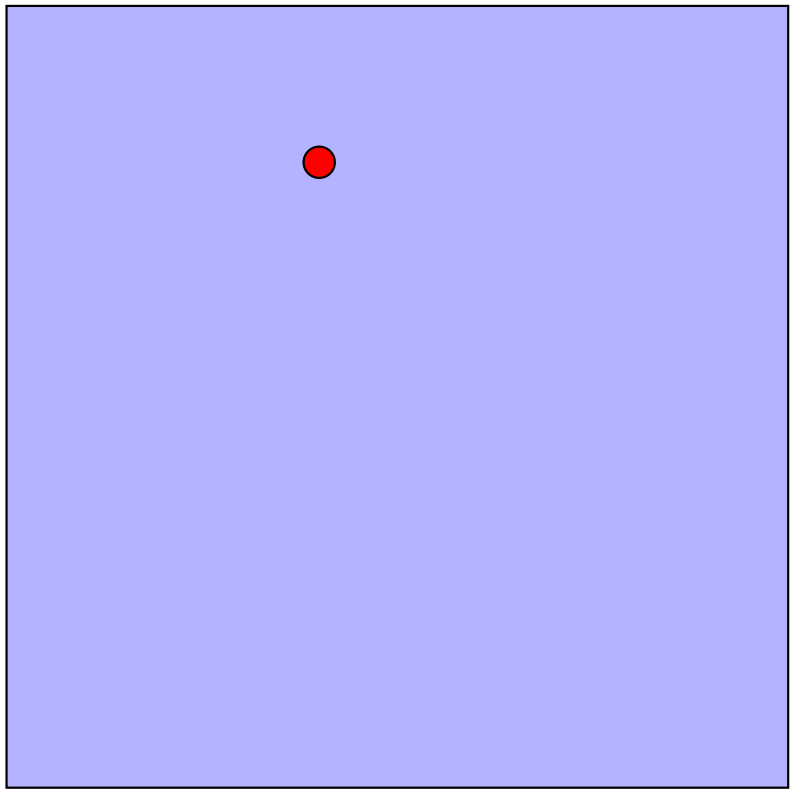
Partición 1



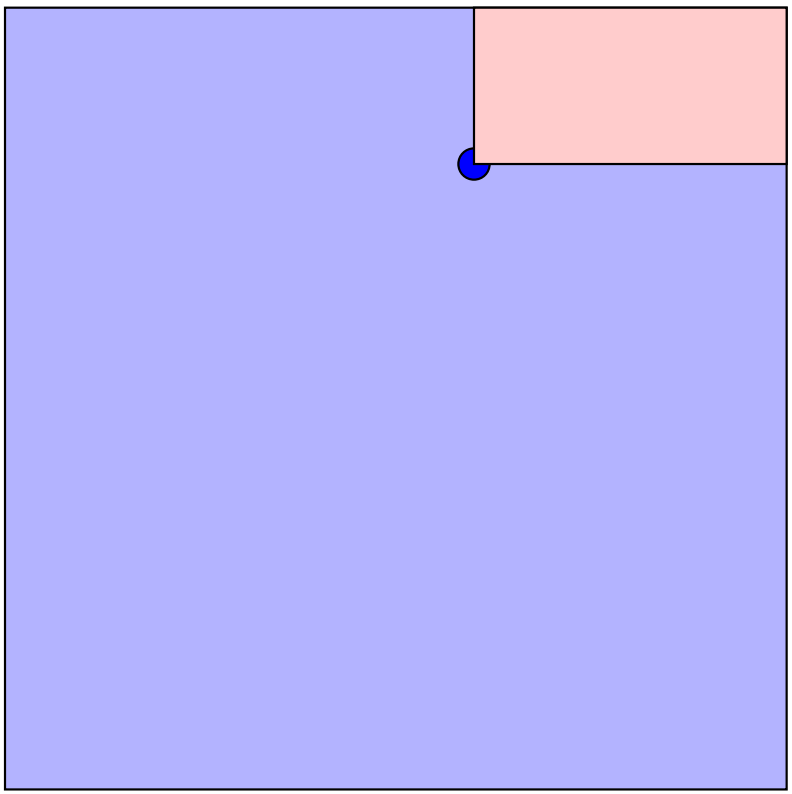
Partición 2



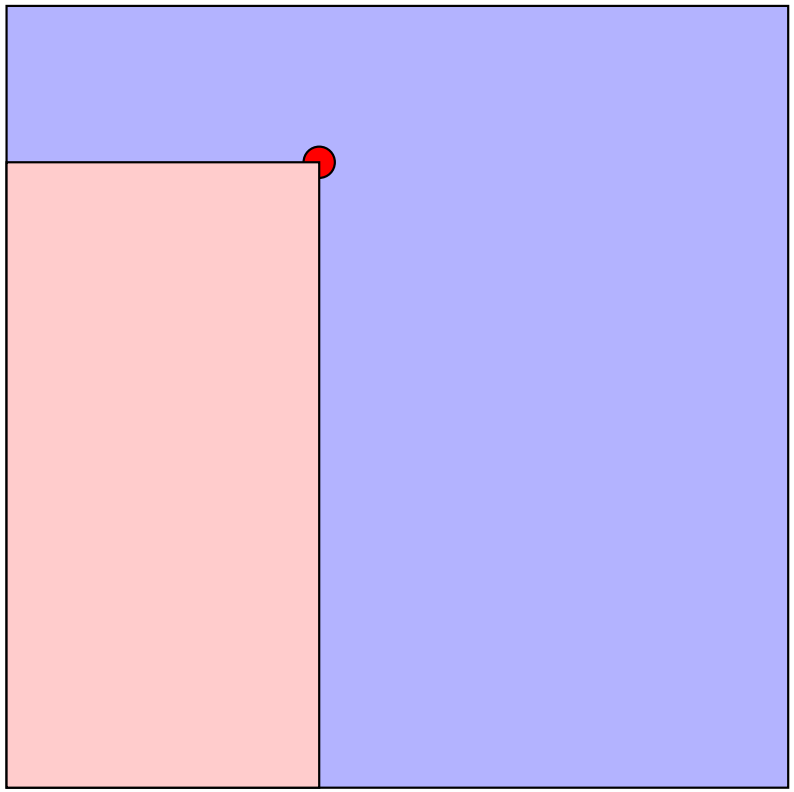
Partición 1



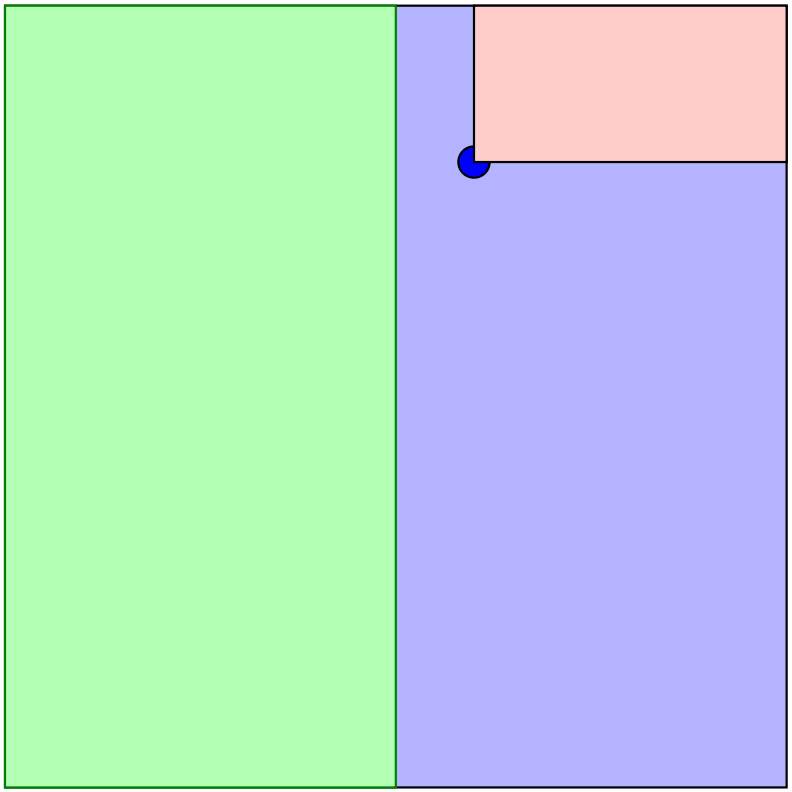
Partición 2



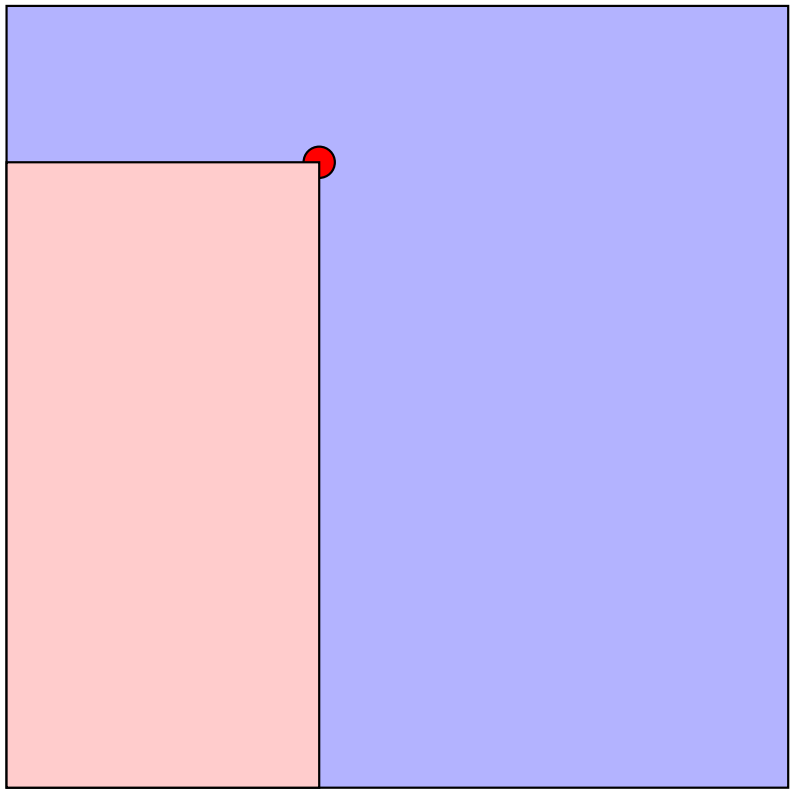
Partición 1



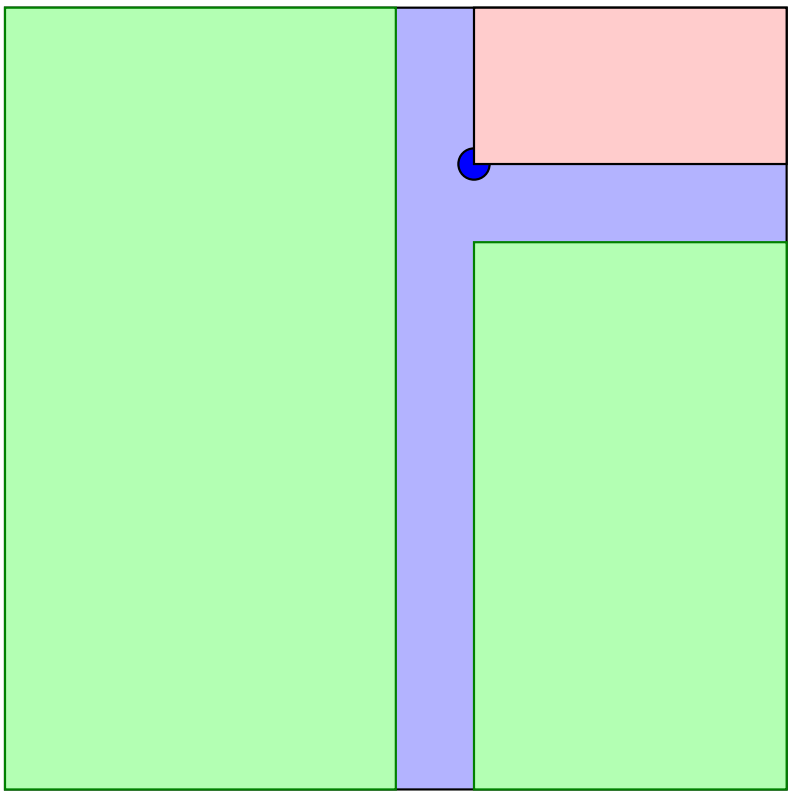
Partición 2



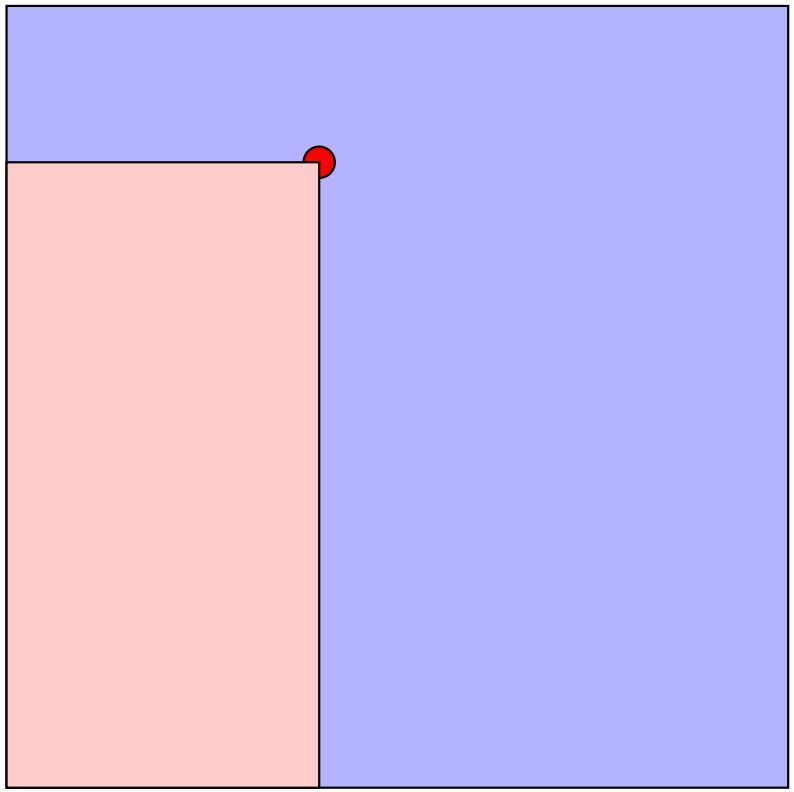
Partición 1



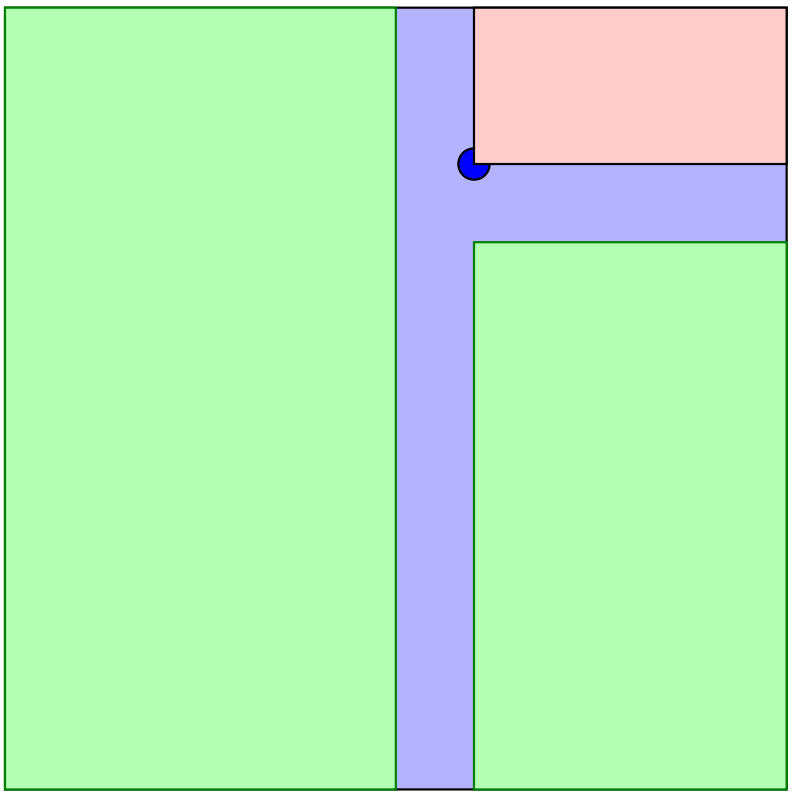
Partición 2



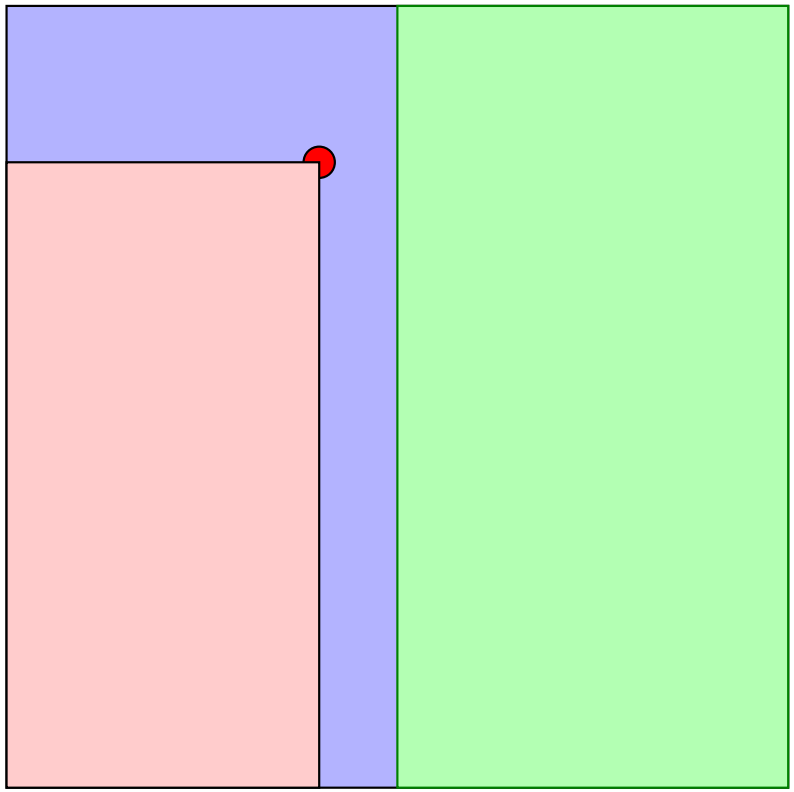
Partición 1



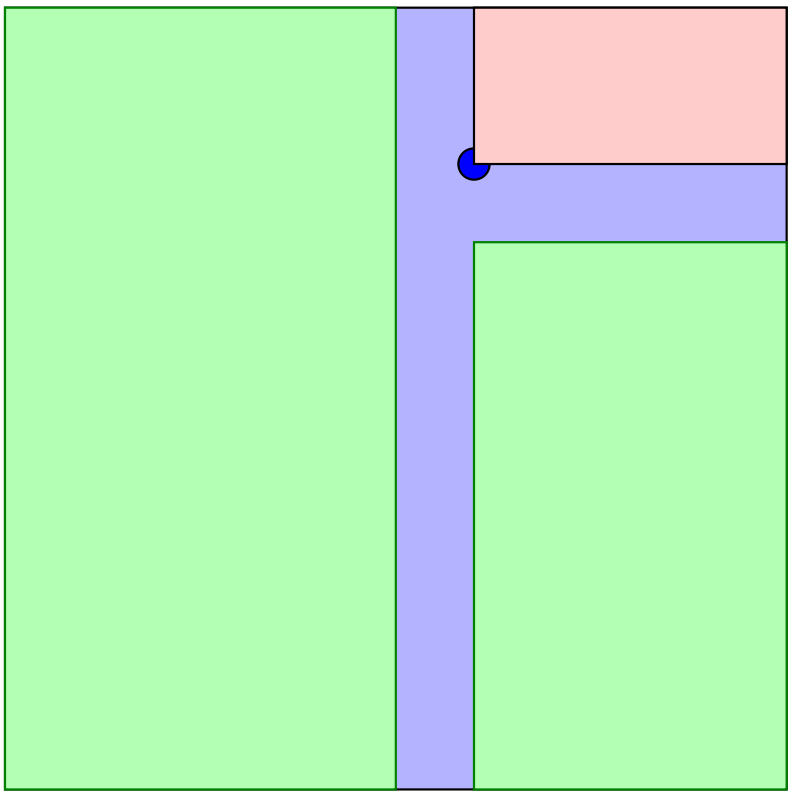
Partición 2



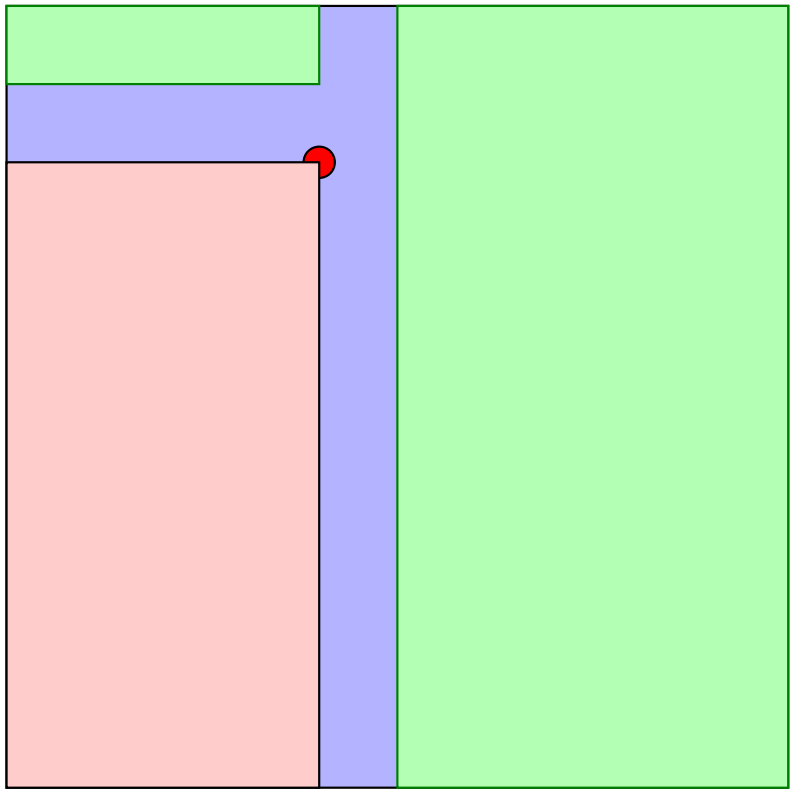
Partición 1



Partición 2



Partición 1



Partición 2

Algoritmo Ruan & Sun

Require: x_{opt} factible

Ensure: x óptimo (P2)

$\Omega = \emptyset$, $\alpha^1 = (0, \ldots, 0)$, $\beta^1 = (u_{11}, \ldots, u_{nk_n})$

$X^1 = \langle \alpha^1, \beta^1 \rangle$ $f_{opt} = c(x_{opt})$.

Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^1.

Calcula x^1 solución óptima asociada a LB_1

Algoritmo Ruan & Sun

Require: x_{opt} factible
Ensure: x óptimo (P2)
$\Omega = \emptyset$, $\alpha^1 = (0, \ldots, 0)$, $\beta^1 = (u_{11}, \ldots, u_{nk_n})$
$X^1 = \langle \alpha^1, \beta^1 \rangle$ $f_{opt} = c(x_{opt})$.
Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^1.
Calcula x^1 solución óptima asociada a LB_1
(Paso 1)
if x^k es factible **then**
  X^k Partición 1, N nuevas cajas
  if $f_{opt} > c(x^k)$ **then**
    $x_{opt} = x^k$
  end if
else
  X^k Partición 2, N nuevas cajas
end if

Algoritmo Ruan & Sun

for all X^k Caja de alguna partición anterior **do**
Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^k.
Calcula x^k solución óptima asociada a LB_1.

Algoritmo Ruan & Sun

for all X^k Caja de alguna partición anterior **do**

Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^k.

Calcula x^k solución óptima asociada a LB_1.

Calcula una cota alternativa LB_2 resolviendo $(P1)$ en X^k.

Algoritmo Ruan & Sun

for all X^k Caja de alguna partición anterior **do**
  Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^k.
  Calcula x^k solución óptima asociada a LB_1.
  Calcula una cota alternativa LB_2 resolviendo $(P1)$ en X^k.
  Sea $LB(X^k) = \text{mín}(LB_1, LB_2)$
  if $LB(X^k) < f_{opt}$ **then**
    $\Omega = \Omega \cup X^k$
  end if
end for

Algoritmo Ruan & Sun

for all X^k Caja de alguna partición anterior **do**
Calcula cota lagrangiana LB_1 de D^1 ó D^2 en X^k.
Calcula x^k solución óptima asociada a LB_1.
Calcula una cota alternativa LB_2 resolviendo $(P1)$ en X^k.
Sea $LB(X^k) = \text{mín}(LB_1, LB_2)$
if $LB(X^k) < f_{opt}$ **then**
$\Omega = \Omega \cup X^k$
end if
end for
if $\Omega = \emptyset$ **then**
x_{opt} es óptimo
else
$LB(X^{k+1}) = \text{mín}\{LB(X) X \in \Omega\}$
x^{k+1} solución óptima asociada a $LB(X^{k+1})$
$\Omega = \Omega \backslash X^{k+1}$
Ir a (Paso 1)

$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$

Relajamos el problema a uno lineal:

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{ij} \geq 1$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Relajado lineal Ruan & Sun:

$$(RP2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq n \log(1 - R_0^{1/n}),$$

$$\sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Relajado lineal Ruan & Sun:

$$(RP2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq n \log(1 - R_0^{1/n}),$$

$$\sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Si $r_{ij} \geq R_0$

$$\sum_{j} x_{ij} \geq 1 \Rightarrow \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

Relajado lineal Ruan & Sun:

$$(RP2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq n \log(1 - R_0^{1/n}),$$

$$\sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

$$0 \leq x_{i,j} \leq u_{i,j}$$

Si $r_{ij} \geq R_0$

$$\sum_{j} x_{ij} \geq 1 \Rightarrow \sum_{j=1}^{k_i} (\log(1 - r_{ij})) x_{ij} \leq \log(1 - R_0)$$

Algoritmo greedy à la Ruan & Sun

$y^0 = (u_{11}, \ldots, u_{nk_n})$, $c^0 = \left(\frac{c_{11}}{-\log(1-r_{11})}, \ldots, \frac{c_{nk_n}}{-\log(1-r_{nk_n})} \right)$
$I = \{(1,1), \ldots, (1,k_1) \ldots, (n,k_n)\}$,
Ordena I por orden decreciente de $c^0_{i,j}$
for all $(i,j) \in I$ **do**
  fiable=true & subsisnovacio=true
  while fiable & subsisnovacio & $y^0_{i,j} > 0$ **do**
    $y^0_{i,j} = y^0_{i,j} - 1$
    if $\sum_k y^0_{ik} < 1$ **then**
      subsisnovacio=false
    end if
    if $R(y^0) < R_0$ **then**
      fiable=false
    end if
    if fiable=false o subsisnovacio=false **then**
      $y^0_{i,j} = y^0_{ij} + 1$
    end if
  end while
end for

Otra condición lineal

Con el algoritmo "greedy", calculamos y^0 factible para $(P2)$.

Otra condición lineal

Con el algoritmo "greedy", calculamos y^0 factible para $(P2)$.
$\sum_{ij} c_{ij} y^0_{ij} = c_G$

Otra condición lineal

Con el algoritmo "greedy", calculamos y^0 factible para $(P2)$. $\sum_{ij} c_{ij} y^0_{ij} = c_G$ El óptimo de $(P2)$, tendrá coste inferior o igual a c_G, por lo que $(P2)$ es equivalente a:

Otra condición lineal

Con el algoritmo "greedy", calculamos y^0 factible para $(P2)$. $\sum_{ij} c_{ij} y^0_{ij} = c_G$ El óptimo de $(P2)$, tendrá coste inferior o igual a c_G, por lo que $(P2)$ es equivalente a:

$$(P2) \qquad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad R(x) = \prod_{i=1}^{n} (1 - \prod_{j=1}^{k_i} (1 - r_{ij})^{x_{ij}}) \geq R_0,$$

$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$
$$\sum_{ij} c_{ij} x_{ij} \leq c_G$$

Relajado lineal de $(P2)$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad 0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_j x_{ij} \geq 1$$
$$\sum_{ij} c_{ij} x_{ij} \leq c_G$$

Relajado lineal de $(P2)$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_j x_{ij} \geq 1 \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

- Queremos calcular un conjunto test para $(LP2)$.

Relajado lineal de $(P2)$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_j x_{ij} \geq 1 \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

- Queremos calcular un conjunto test para $(LP2)$.
- Usar el conjunto test para resolver $(P2)$.

Conjunto test

Un problema de programación lineal entera se resuelve en general: Consideramos problemas con coste $\mathbf{c}$ y matriz A. La fibra de un IP se obtiene fijando $\mathbf{b}$ en $A\mathbf{x} = \mathbf{b}$.

Conjunto test

Un problema de programación lineal entera se resuelve en general: Consideramos problemas con coste $\mathbf{c}$ y matriz A. La fibra de un IP se obtiene fijando $\mathbf{b}$ en $A\mathbf{x} = \mathbf{b}$.

Definición conjunto test

Un conjunto $\mathcal{G}_{>_c} \subset \mathbb{Z}^N$ es un conjunto test de una familia de IP con matriz A y coste $\mathbf{c}$ si

- para cada punto no óptimo α en cada fibra de IP existe $g \in \mathcal{G}_{>_c}$ tal que $\alpha - g$ es un punto factible en la fibra y $\alpha >_c \alpha - g$,

Conjunto test

Un problema de programación lineal entera se resuelve en general: Consideramos problemas con coste $\mathbf{c}$ y matriz A. La fibra de un IP se obtiene fijando $\mathbf{b}$ en $A\mathbf{x} = \mathbf{b}$.

Definición conjunto test

Un conjunto $\mathcal{G}_{>_c} \subset \mathbb{Z}^N$ es un conjunto test de una familia de IP con matriz A y coste $\mathbf{c}$ si

- para cada punto no óptimo α en cada fibra de IP existe $g \in \mathcal{G}_{>_c}$ tal que $\alpha - g$ es un punto factible en la fibra y $\alpha >_c \alpha - g$,
- para el punto óptimo β en una fibra de IP, $\beta - g$ es no factible para todo $g \in \mathcal{G}_{>_c}$.

Propiedades de un conjunto test

- Un conjunto test proporciona un método para resolver un IP dado un punto factible.
- En cada paso, o existe un elemento del conjunto test que mejora al actual, o no hay mejora (estamos en el óptimo).
- El proceso finaliza por la hipótesis de coste acotado.

Entrada al laberinto

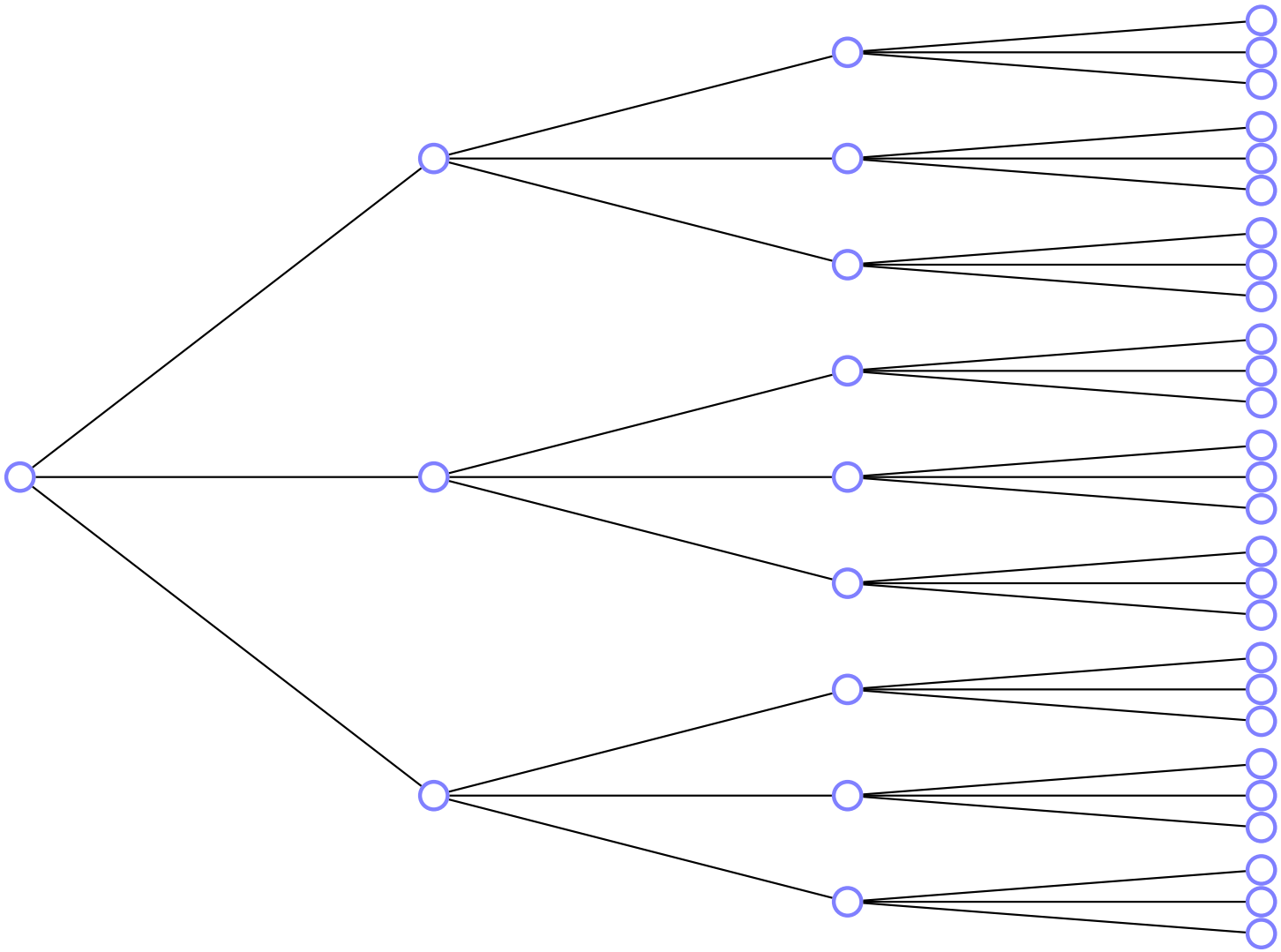
Dado **b**, se construye un grafo dirigido a partir de $\mathcal{G}_{>_c}$ (esqueleto de **b**), que verifica

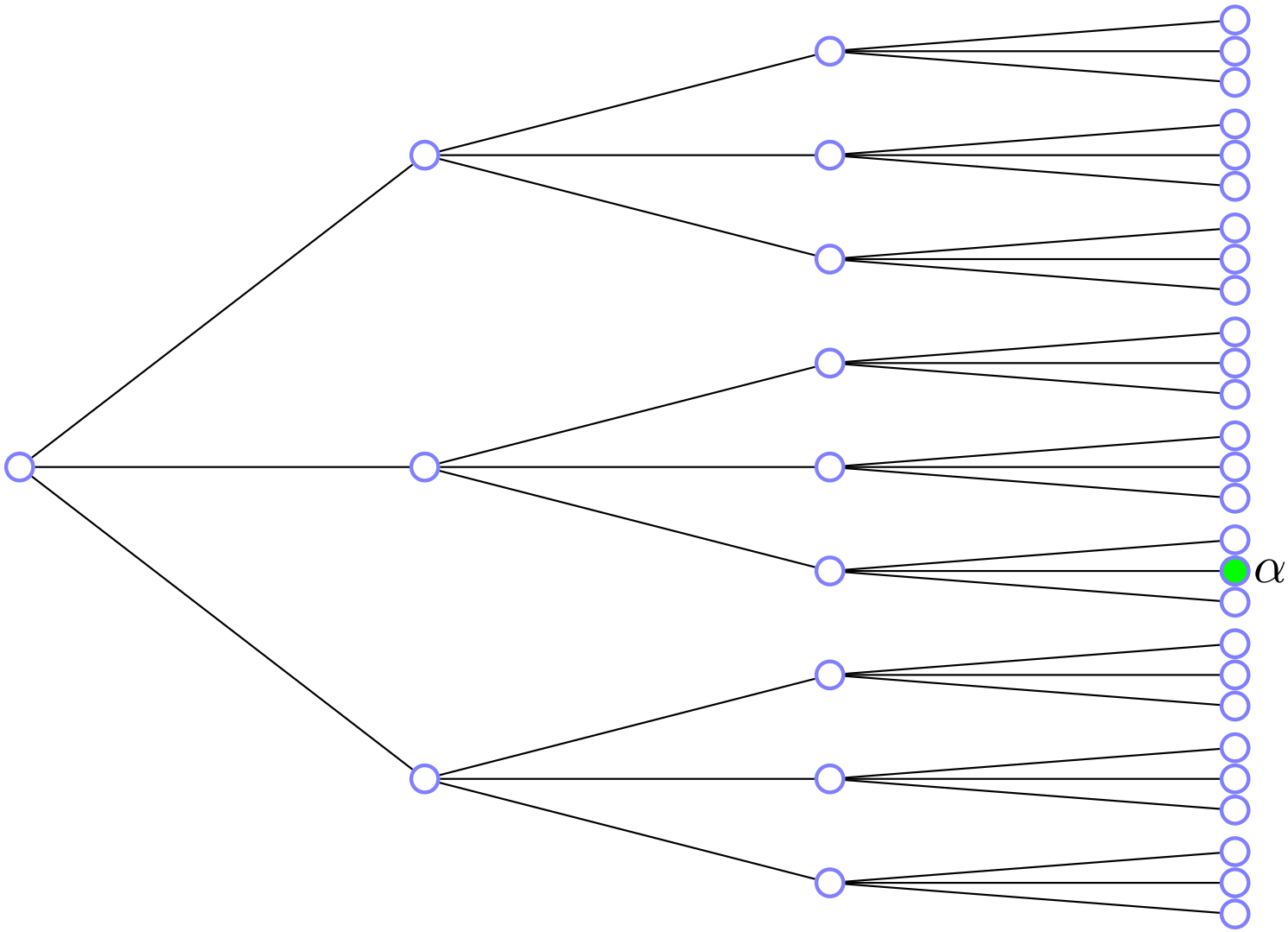
- existe un grafo dirigido de cada punto no óptimo, factible α al único óptimo β,

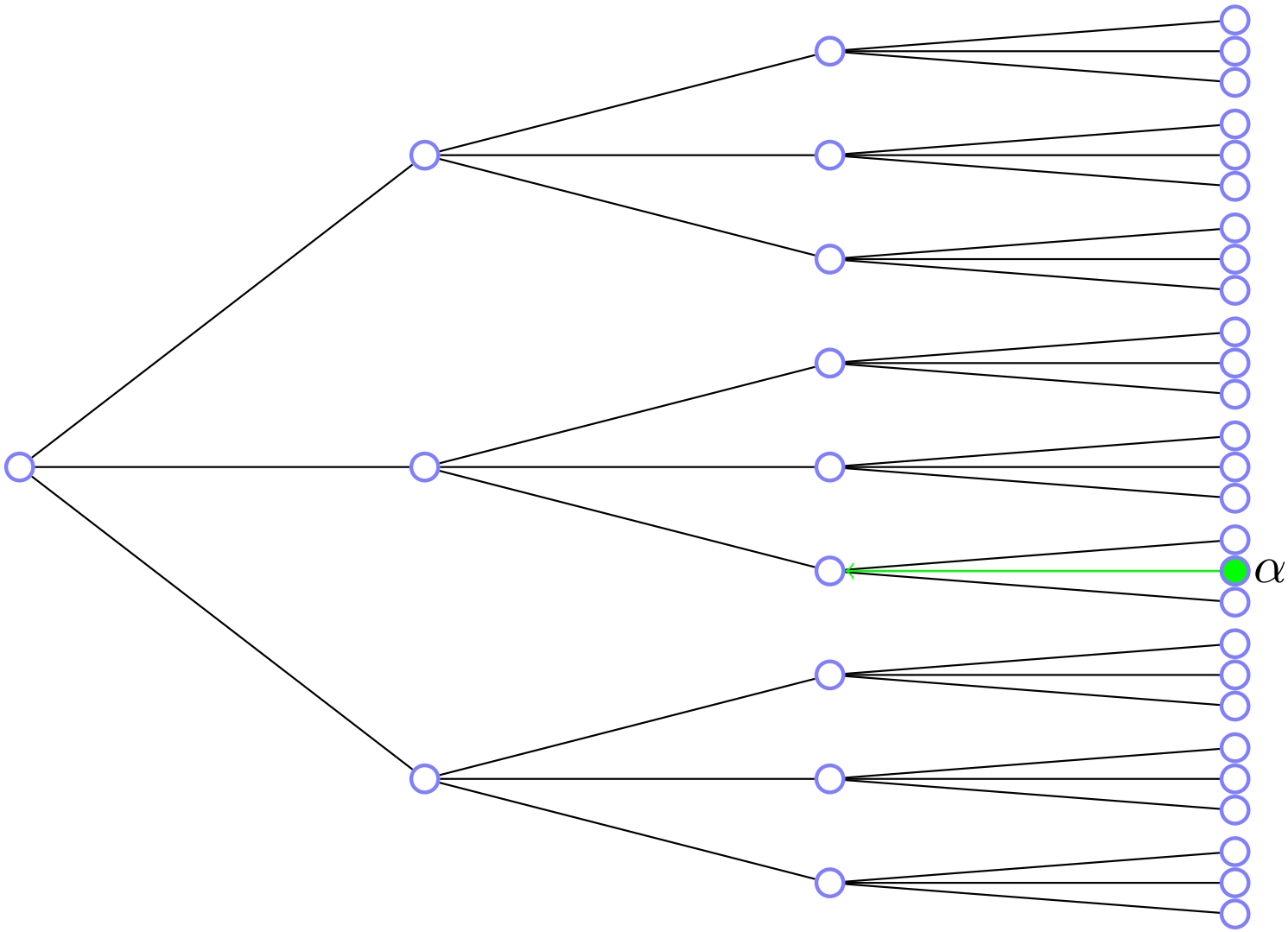
Entrada al laberinto

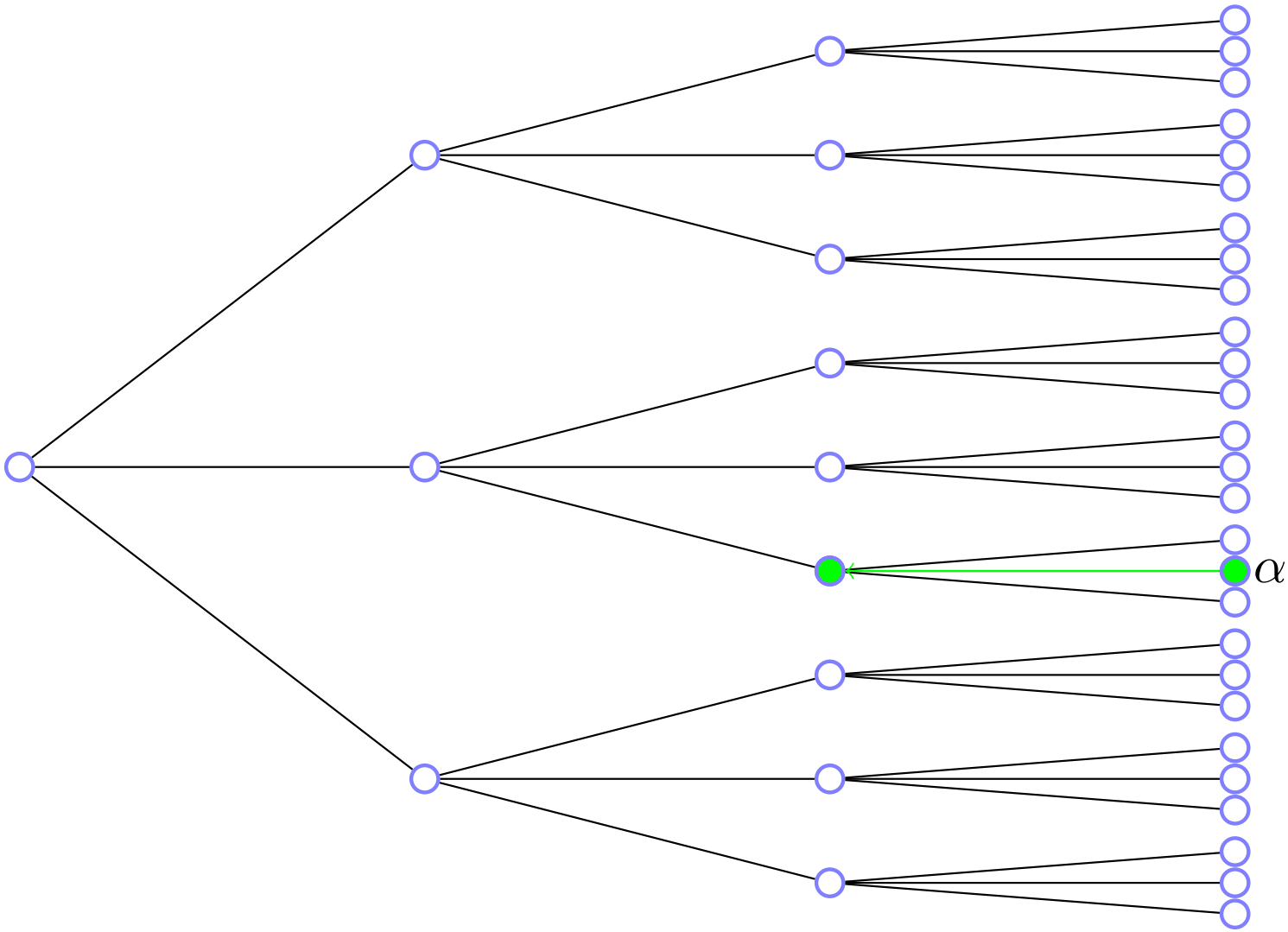
Dado **b**, se construye un grafo dirigido a partir de $\mathcal{G}_{>_c}$ (esqueleto de **b**), que verifica

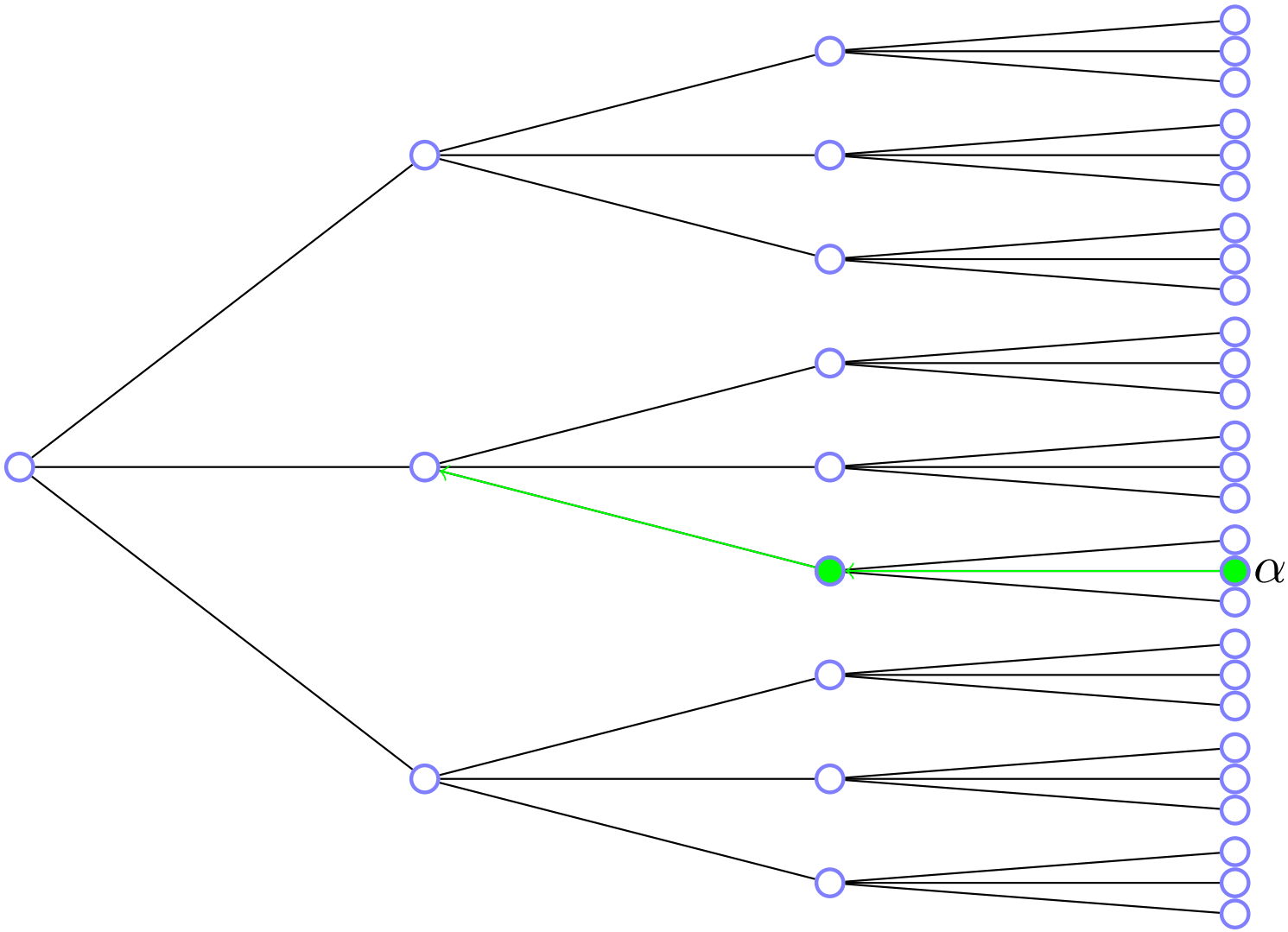
- existe un grafo dirigido de cada punto no óptimo, factible α al único óptimo β,
- la función objetivo sobre puntos contiguos decrece de forma monótona desde α a β.

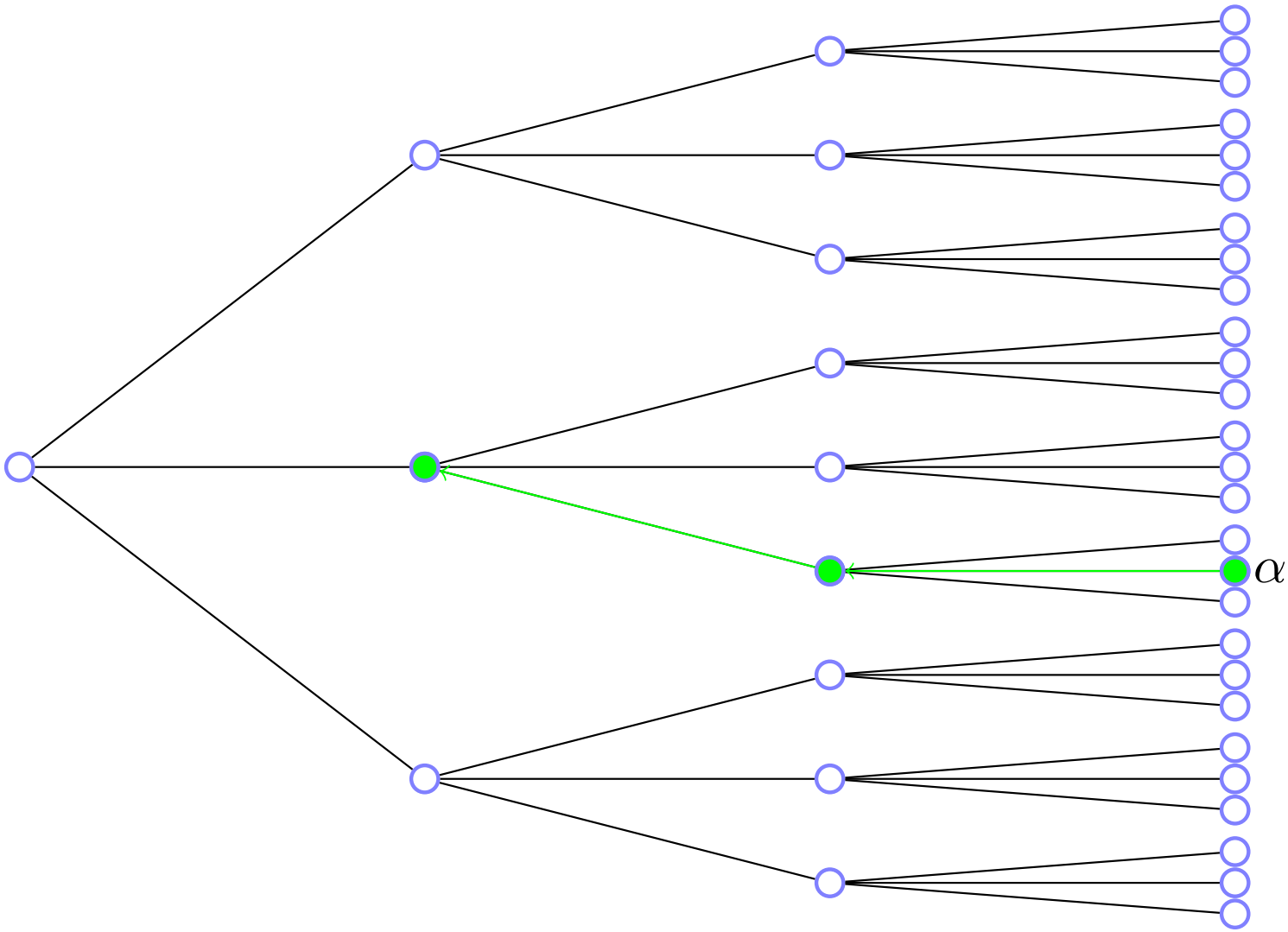


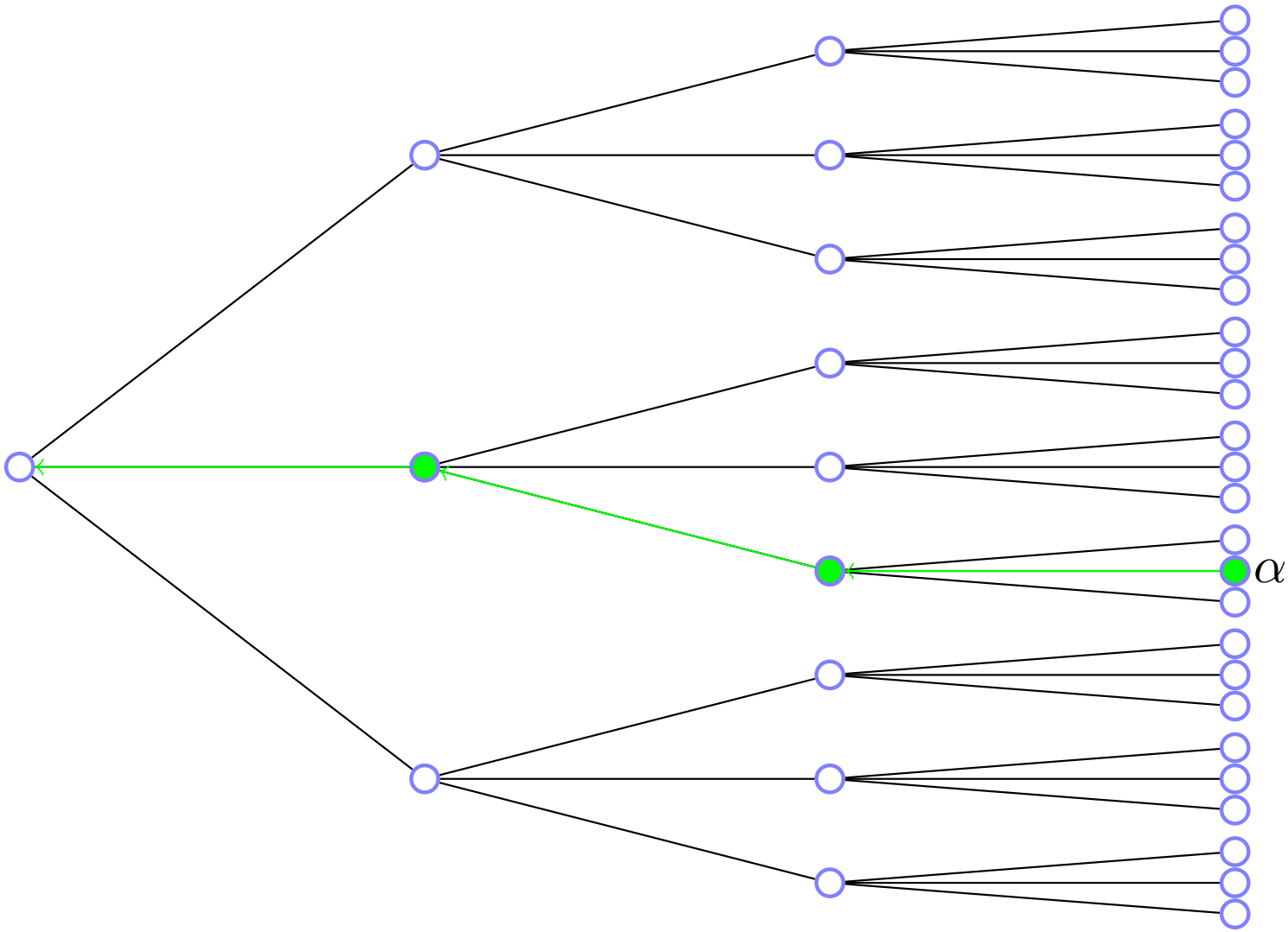

α

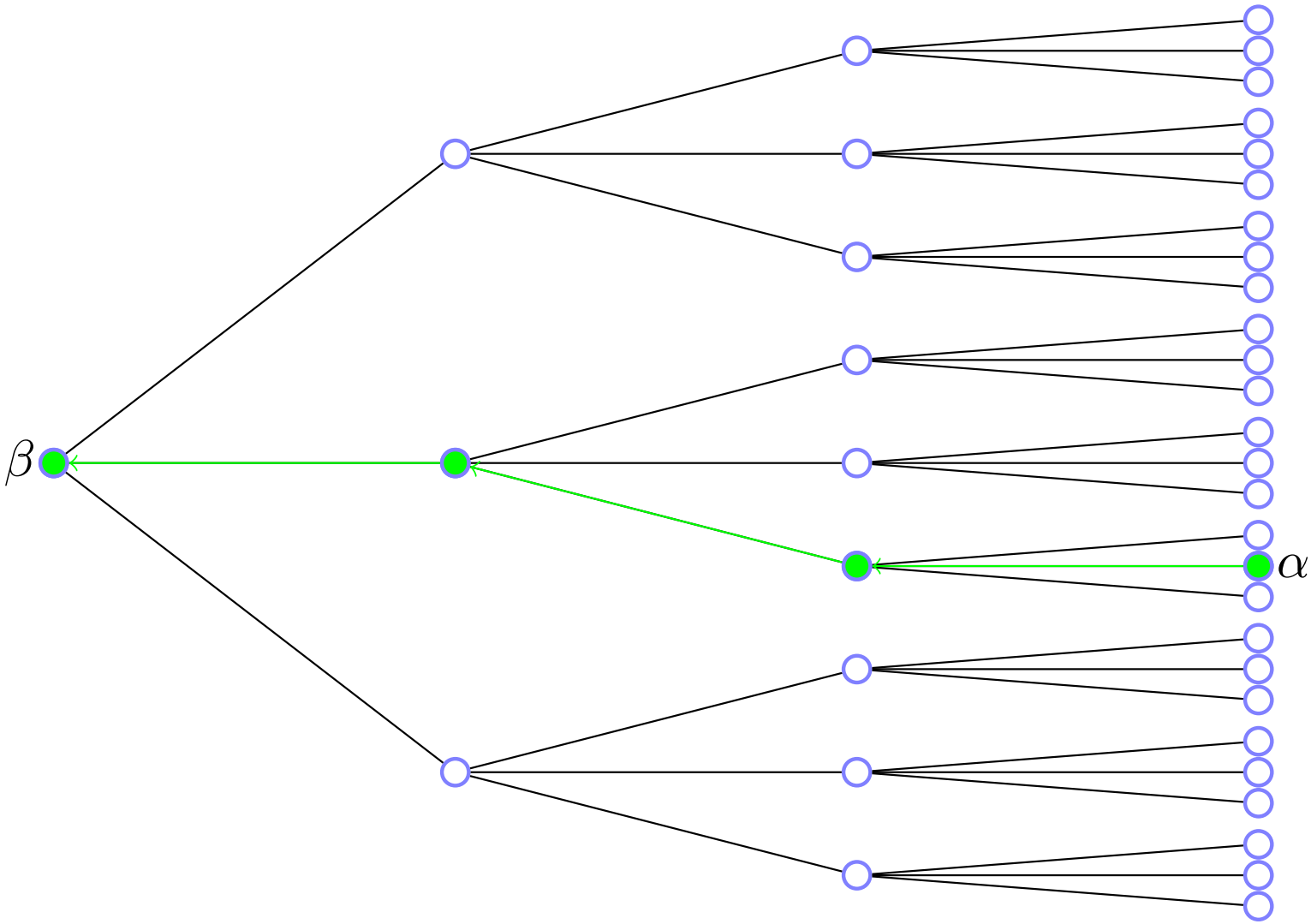

α


α


α


α


α


β
α

Salida del laberinto

- El *esqueleto reverso* $\mathcal{G}'_{>_c}$ se obtiene dando la vuelta a las flechas del grafo anterior.

Salida del laberinto

- El *esqueleto reverso* $\mathcal{G}'_{>_c}$ se obtiene dando la vuelta a las flechas del grafo anterior.
- Existe un grafo dirigido desde el óptimo a un punto factible de la fibra, con la función objetivo en decrecimiento monótono.

Salida del laberinto

- El *esqueleto reverso* $\mathcal{G}'_{>_c}$ se obtiene dando la vuelta a las flechas del grafo anterior.
- Existe un grafo dirigido desde el óptimo a un punto factible de la fibra, con la función objetivo en decrecimiento monótono.
- $P(\alpha)$: camino en el esqueleto reverso de la fibra desde el punto óptimo β a α.

Resolviendo ($P2$)

- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G

Resolviendo ($P2$)

- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).

Resolviendo ($P2$)

- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).

Resolviendo ($P2$)

- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).
- Si β no es fiable, usamos el esqueleto reverso $\mathcal{G}'_{>_c}$.

Resolviendo ($P2$)

- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).
- Si β no es fiable, usamos el esqueleto reverso $\mathcal{G}'_{>_c}$.
- Para cada γ punto obtenido mediante $\mathcal{G}'_{>_c}$:

Resolviendo ($P2$)

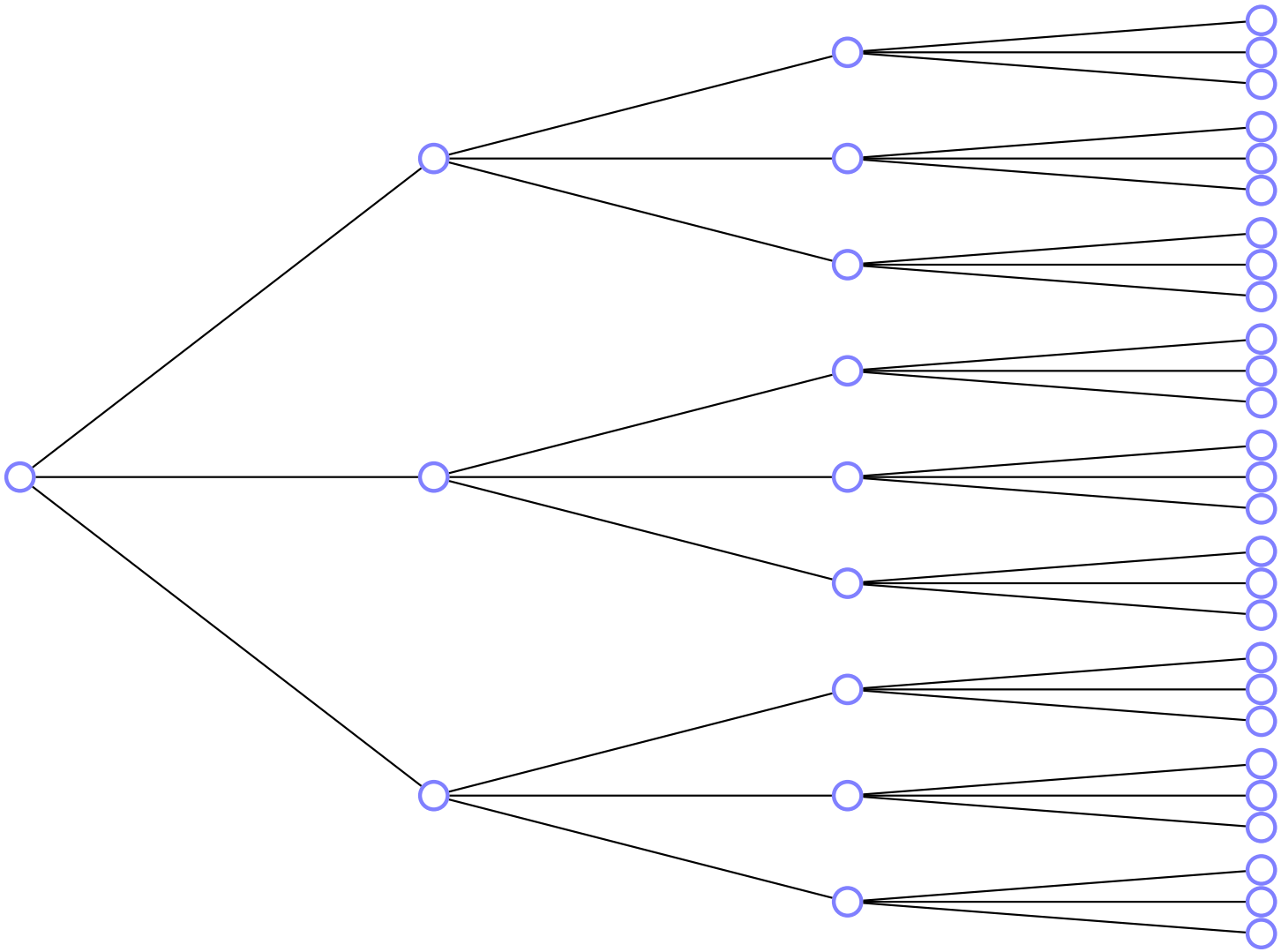
- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).
- Si β no es fiable, usamos el esqueleto reverso $\mathcal{G}'_{>_c}$.
- Para cada γ punto obtenido mediante $\mathcal{G}'_{>_c}$:
  - Si $c(\gamma) > c_G$, podamos la rama.

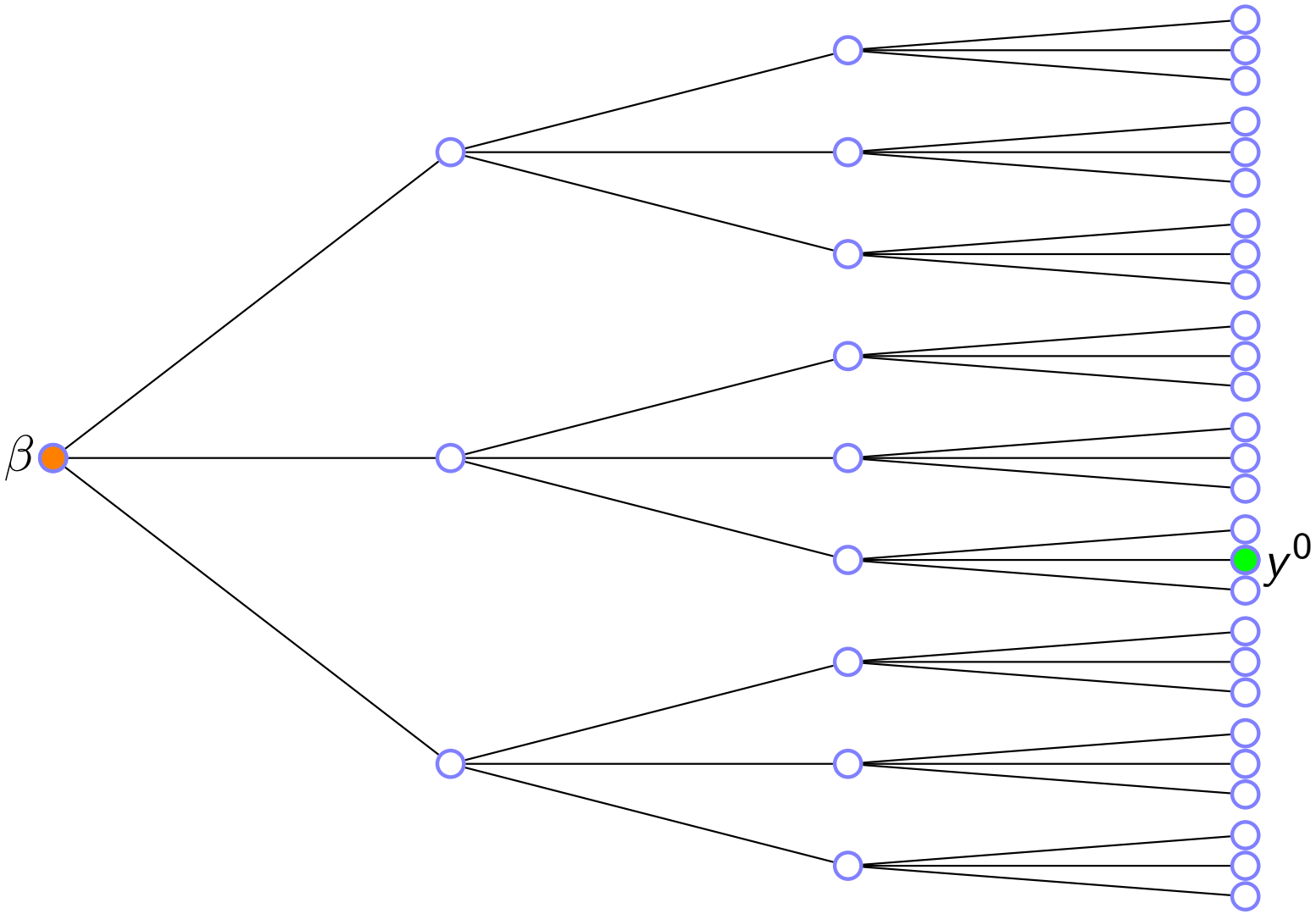
Resolviendo ($P2$)

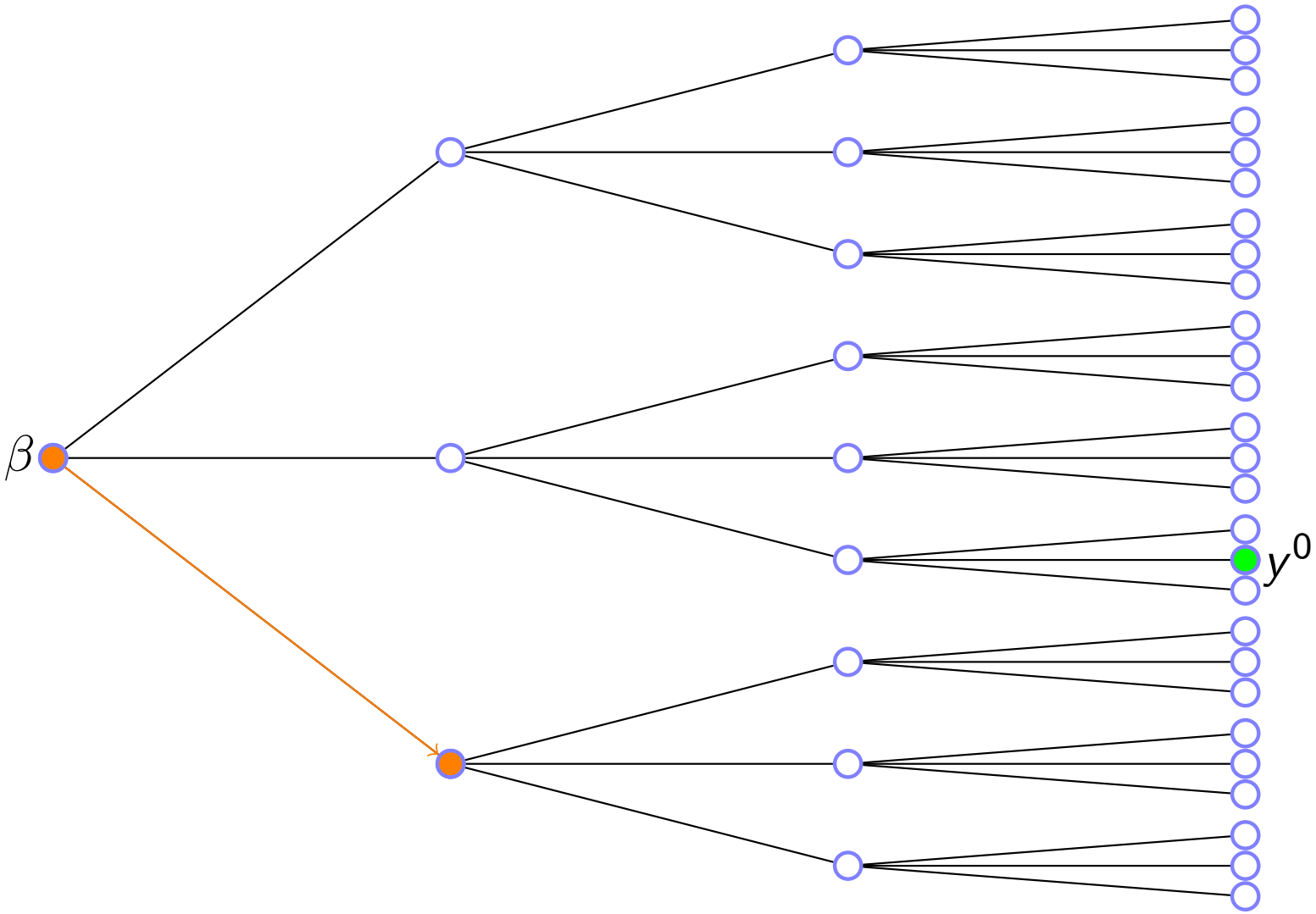
- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).
- Si β no es fiable, usamos el esqueleto reverso $\mathcal{G}'_{>_c}$.
- Para cada γ punto obtenido mediante $\mathcal{G}'_{>_c}$:
  - Si $c(\gamma) > c_G$, podamos la rama.
  - Si $\gamma_i < 0$ podamos la rama.

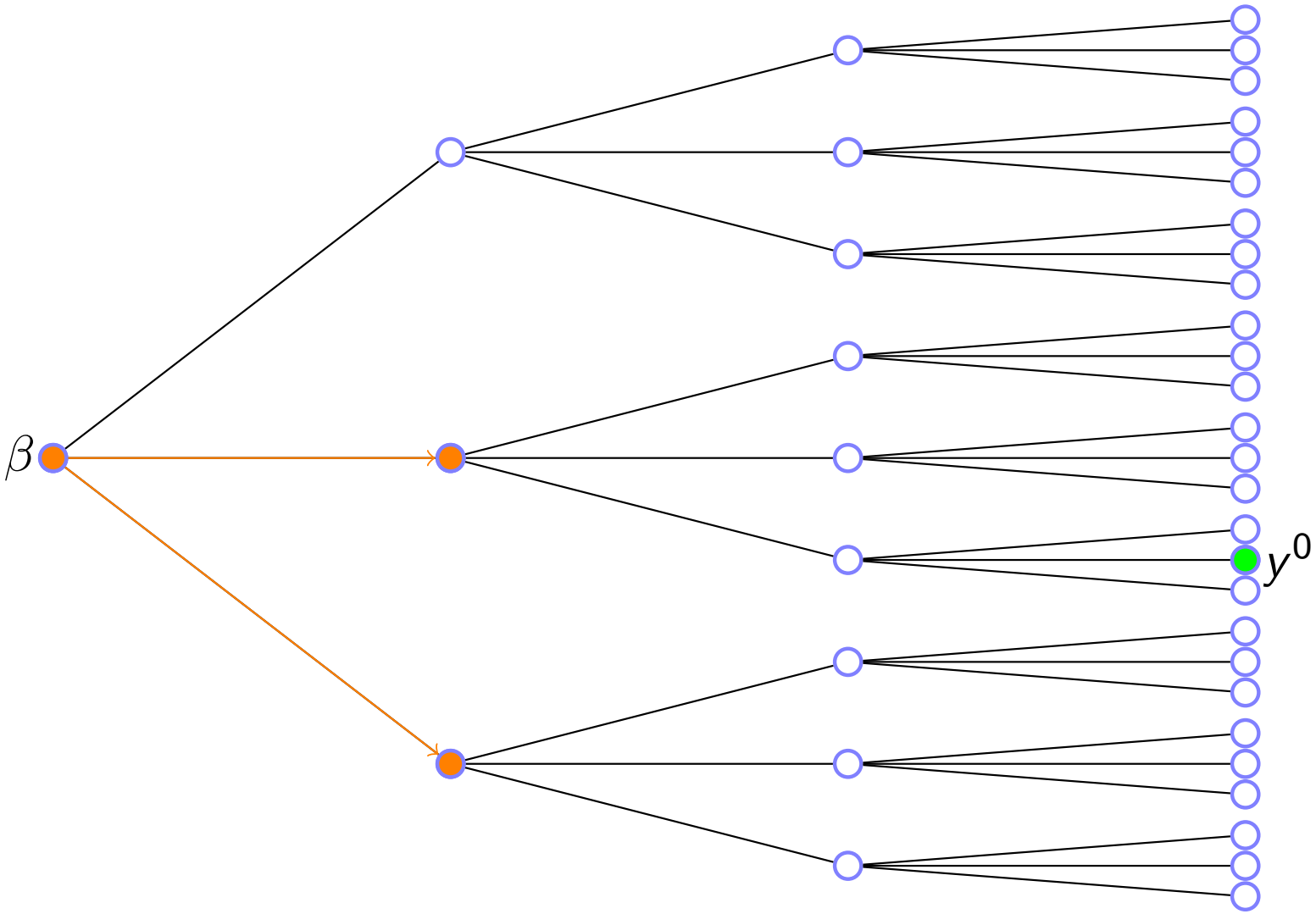
Resolviendo ($P2$)

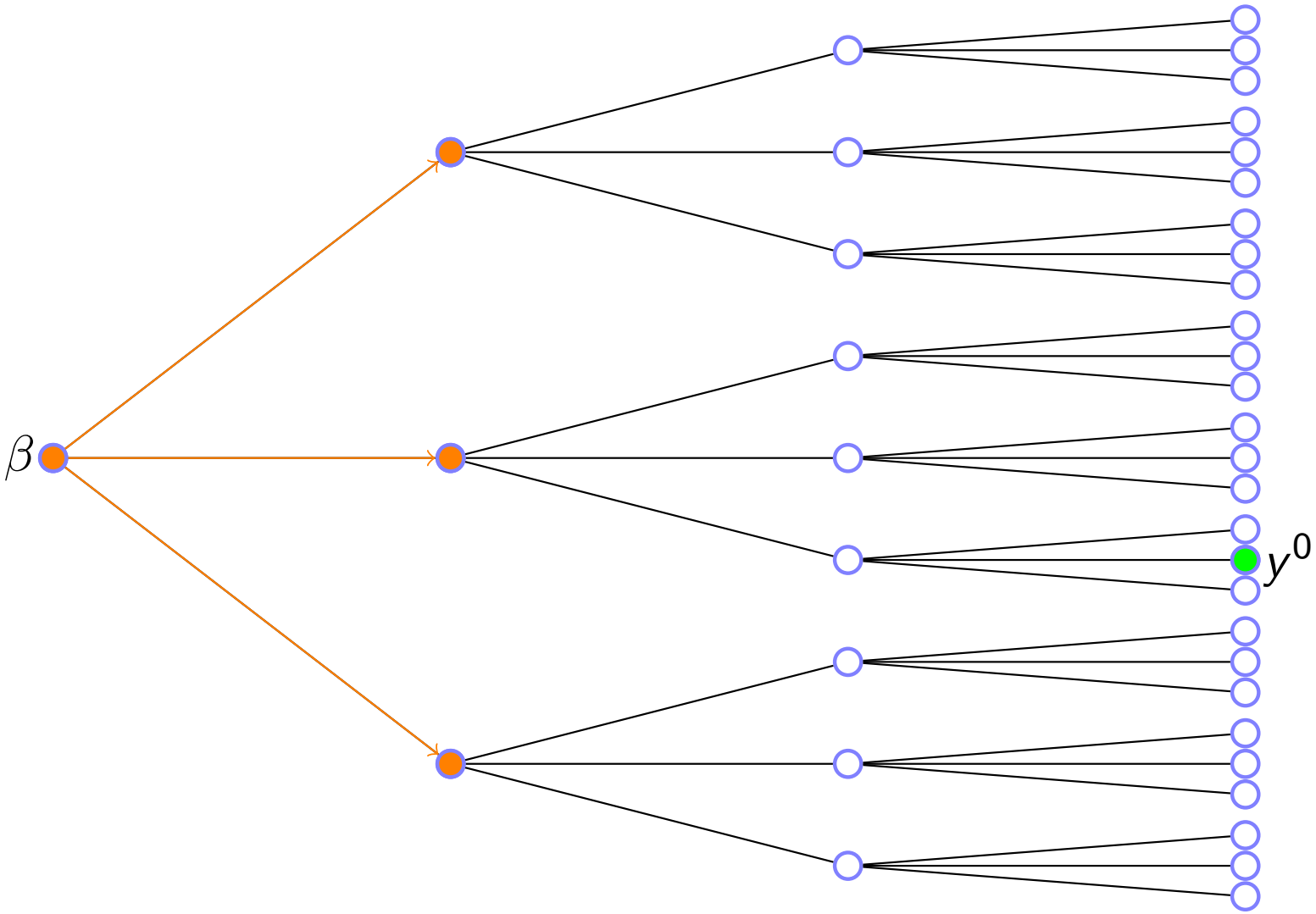
- Partimos del punto y^0, factible para ($P2$), obtenido mediante el algoritmo "greedy". Con coste c_G
- Usando $\mathcal{G}_{>_c}$, calculamos β óptimo de ($LP2$).
- Si β es fiable, es óptimo de ($P2$).
- Si β no es fiable, usamos el esqueleto reverso $\mathcal{G}'_{>_c}$.
- Para cada γ punto obtenido mediante $\mathcal{G}'_{>_c}$:
  - Si $c(\gamma) > c_G$, podamos la rama.
  - Si $\gamma_i < 0$ podamos la rama.
  - Si γ es fiable, y $c(\gamma) < c_G$, actualizamos c_G e y^0.

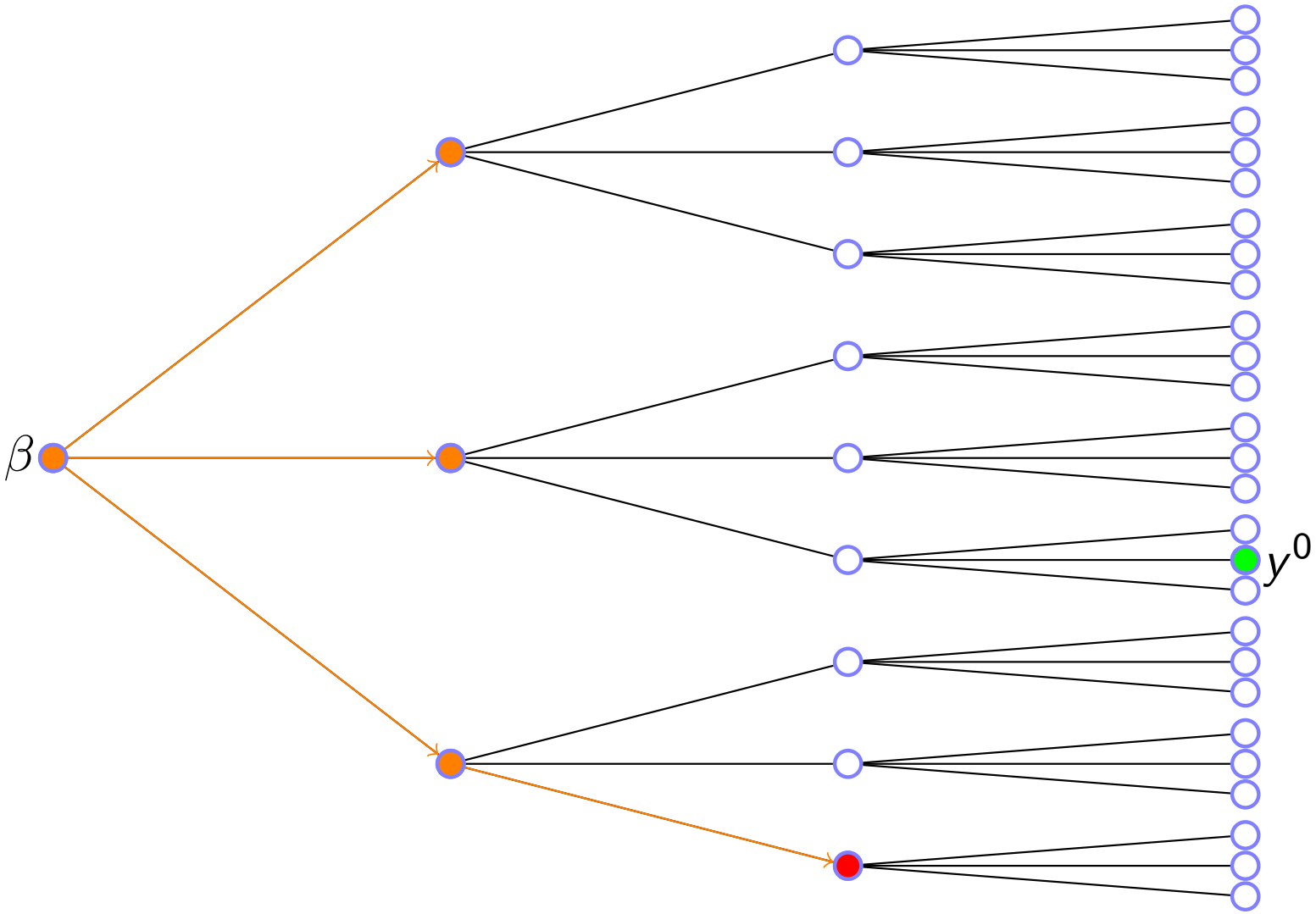


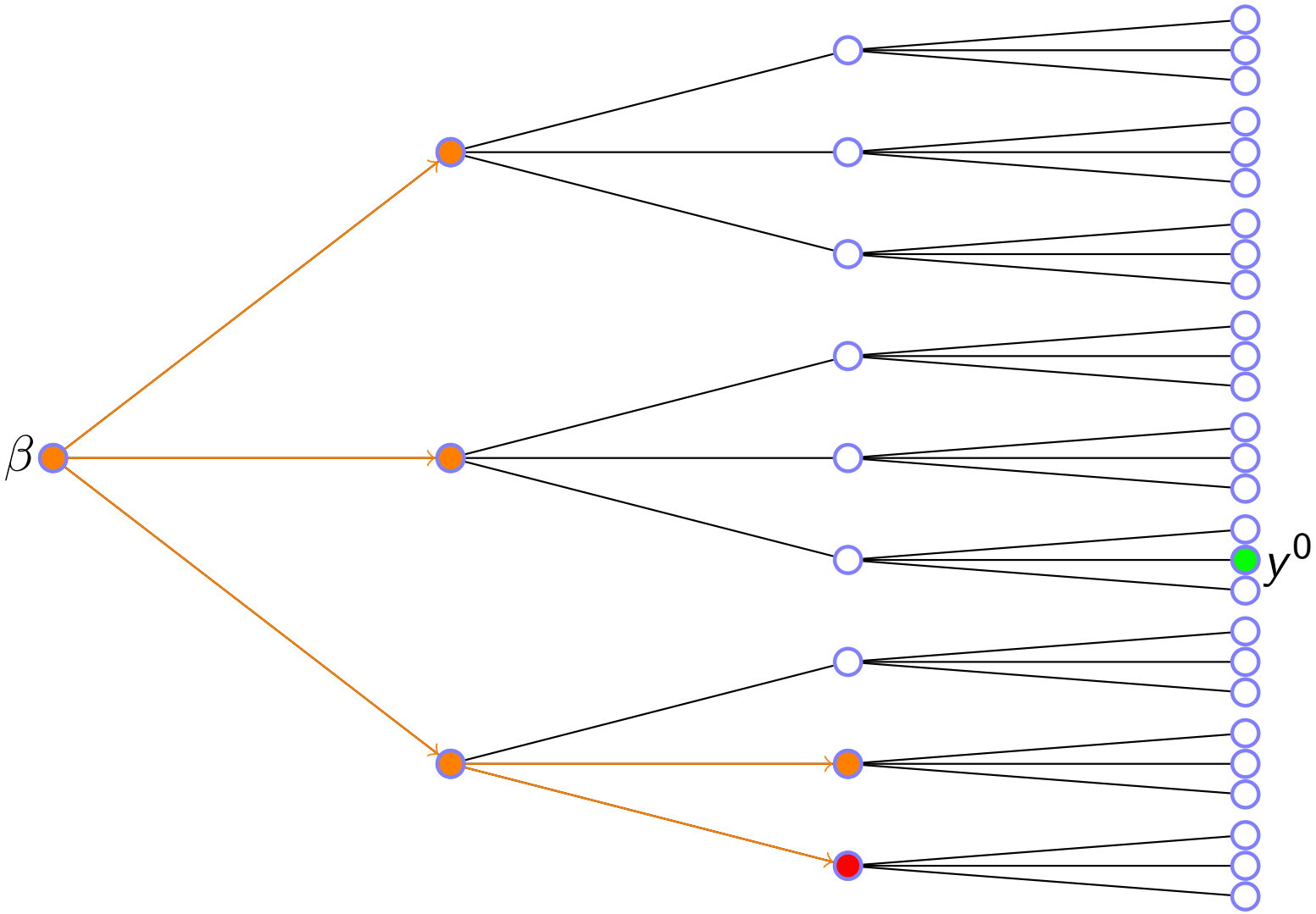

β
y^0

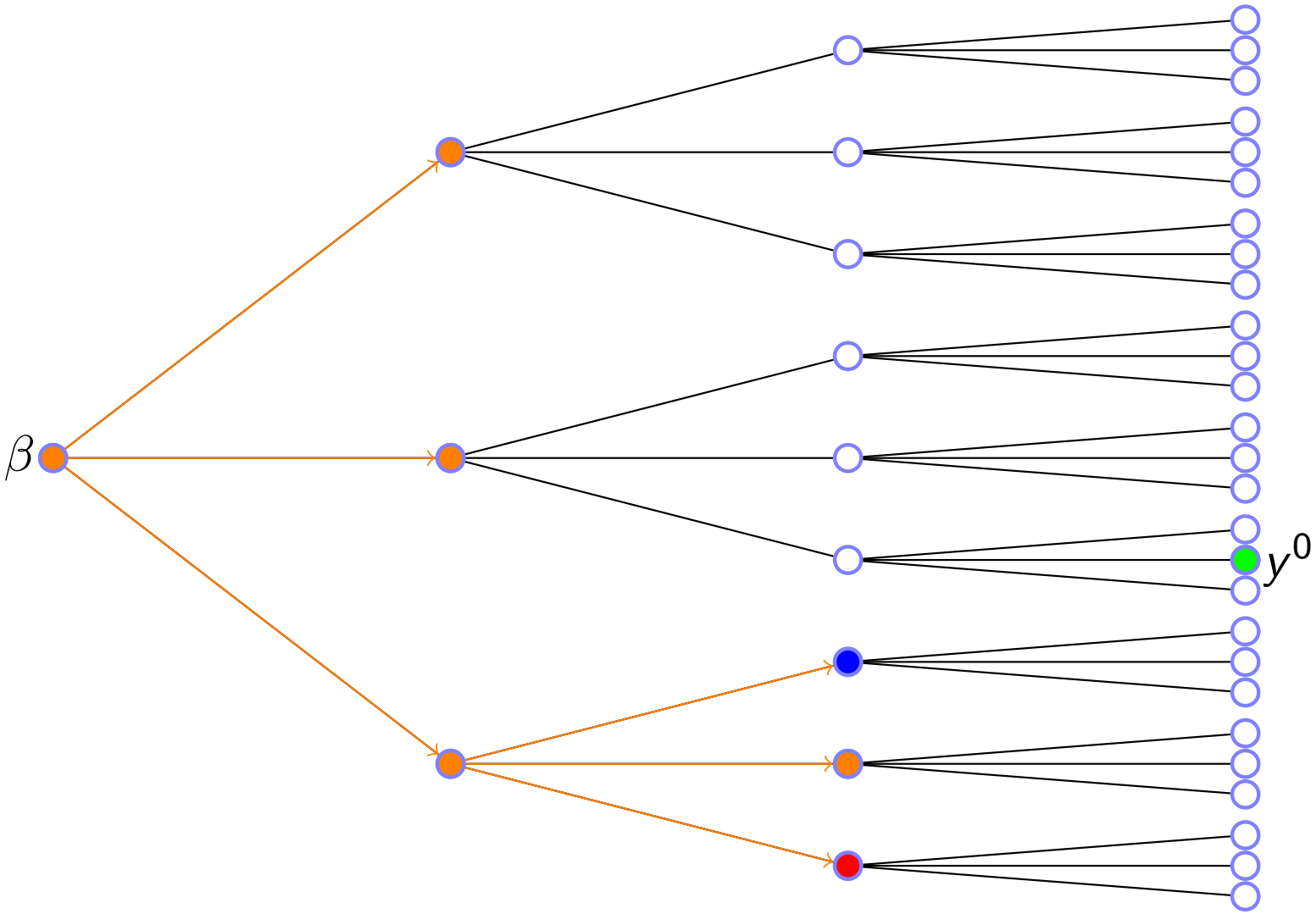

β
y^0

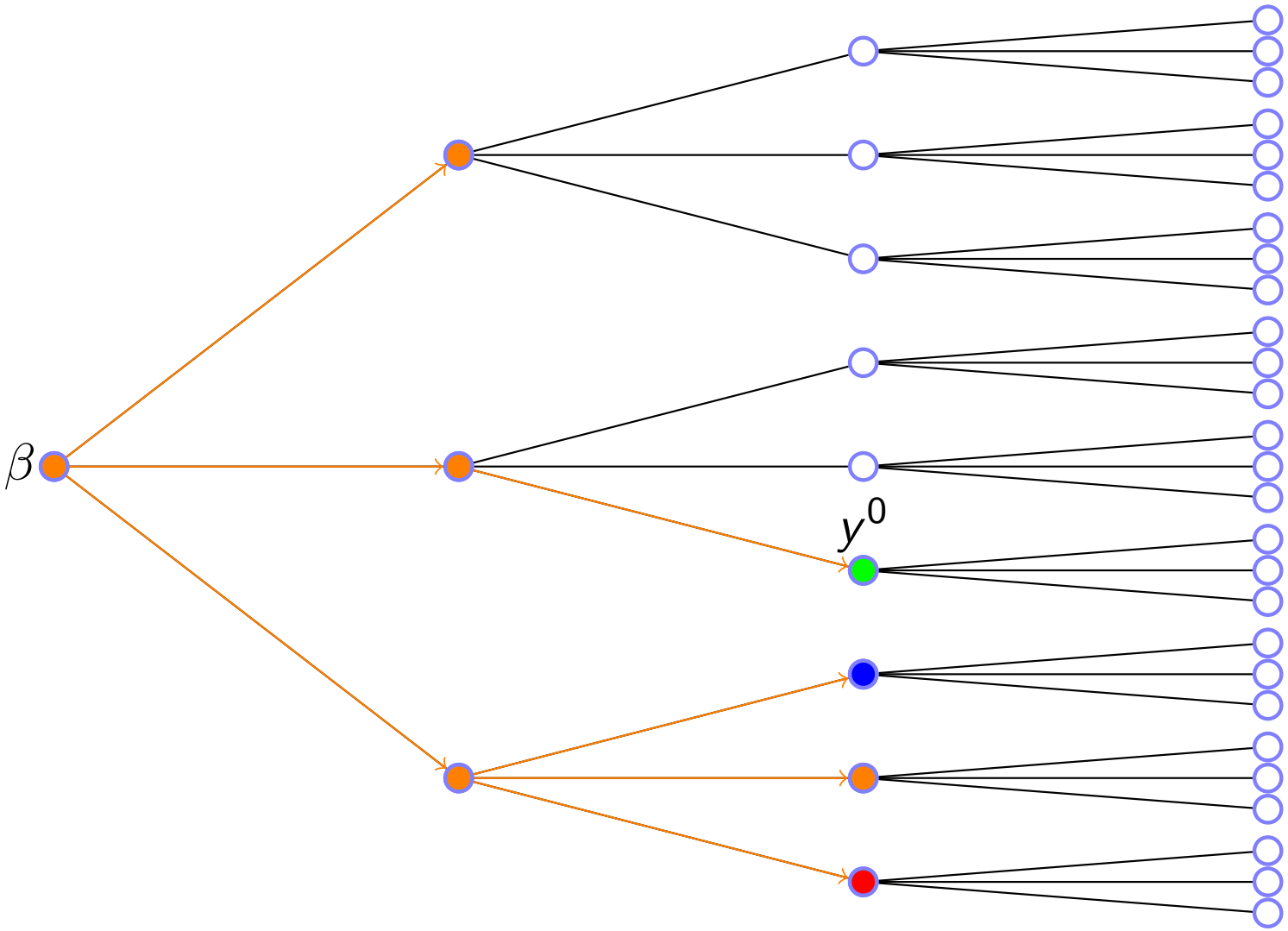

β
y^0

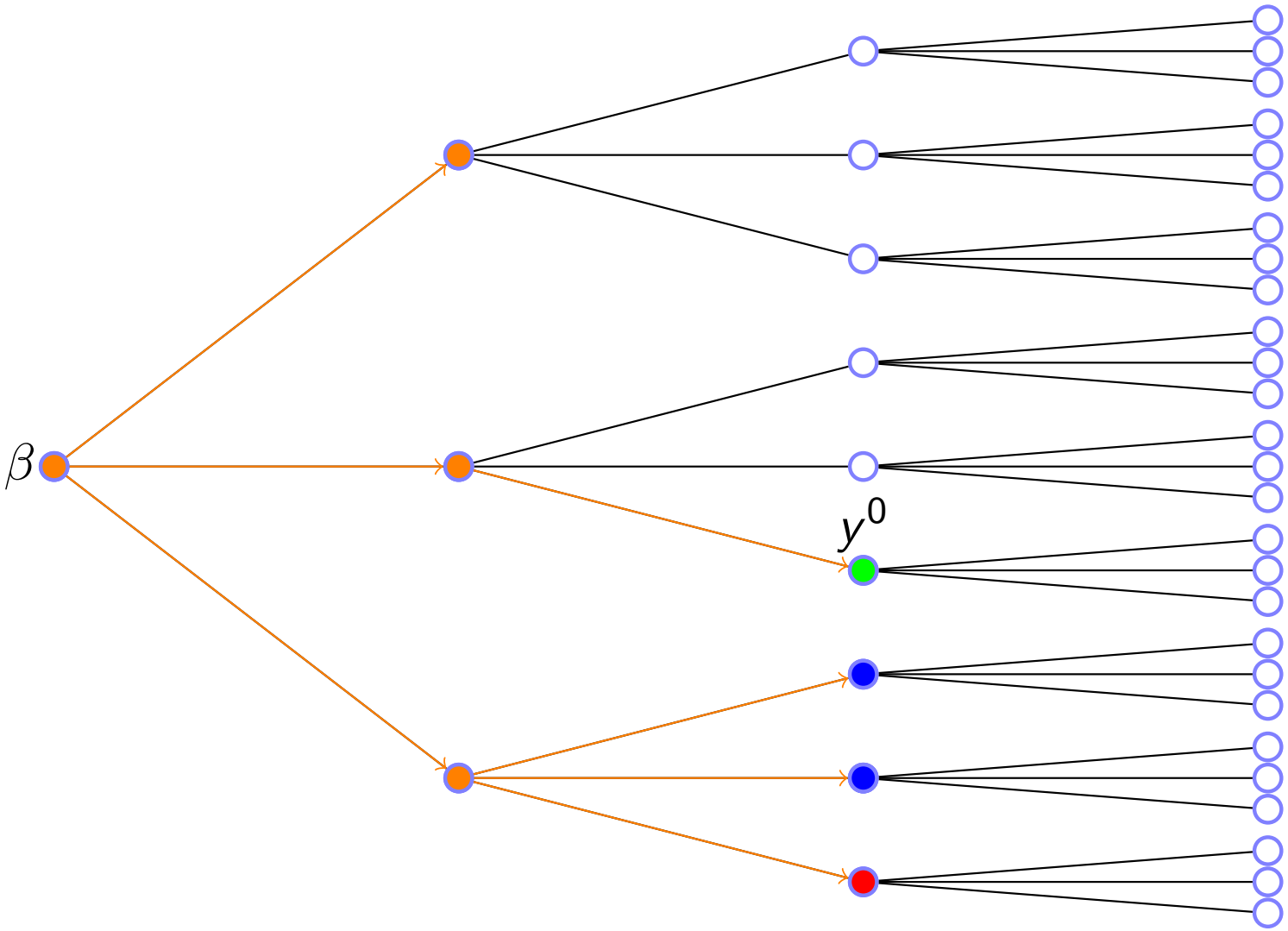

β
y^0

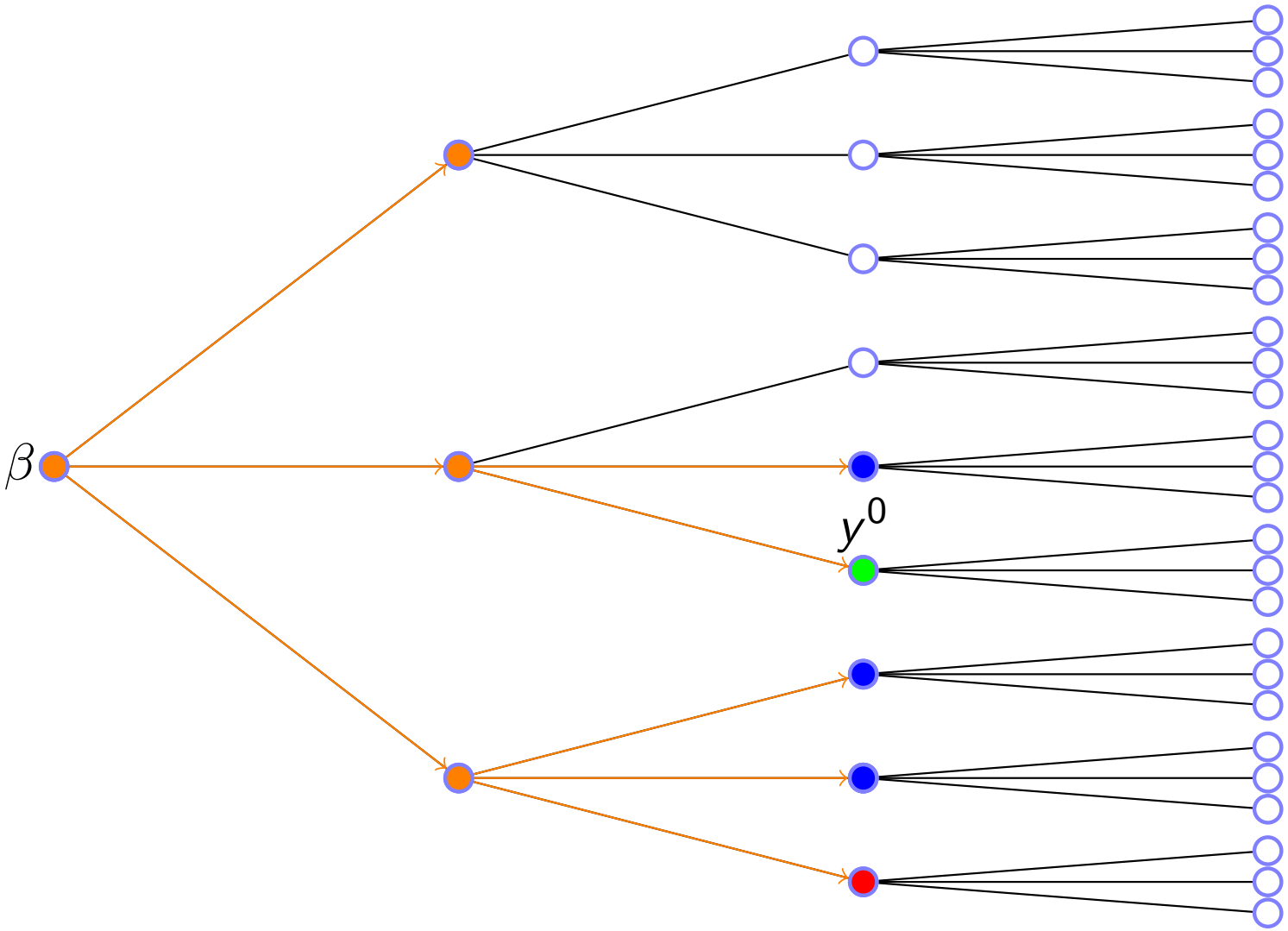

β
y^0

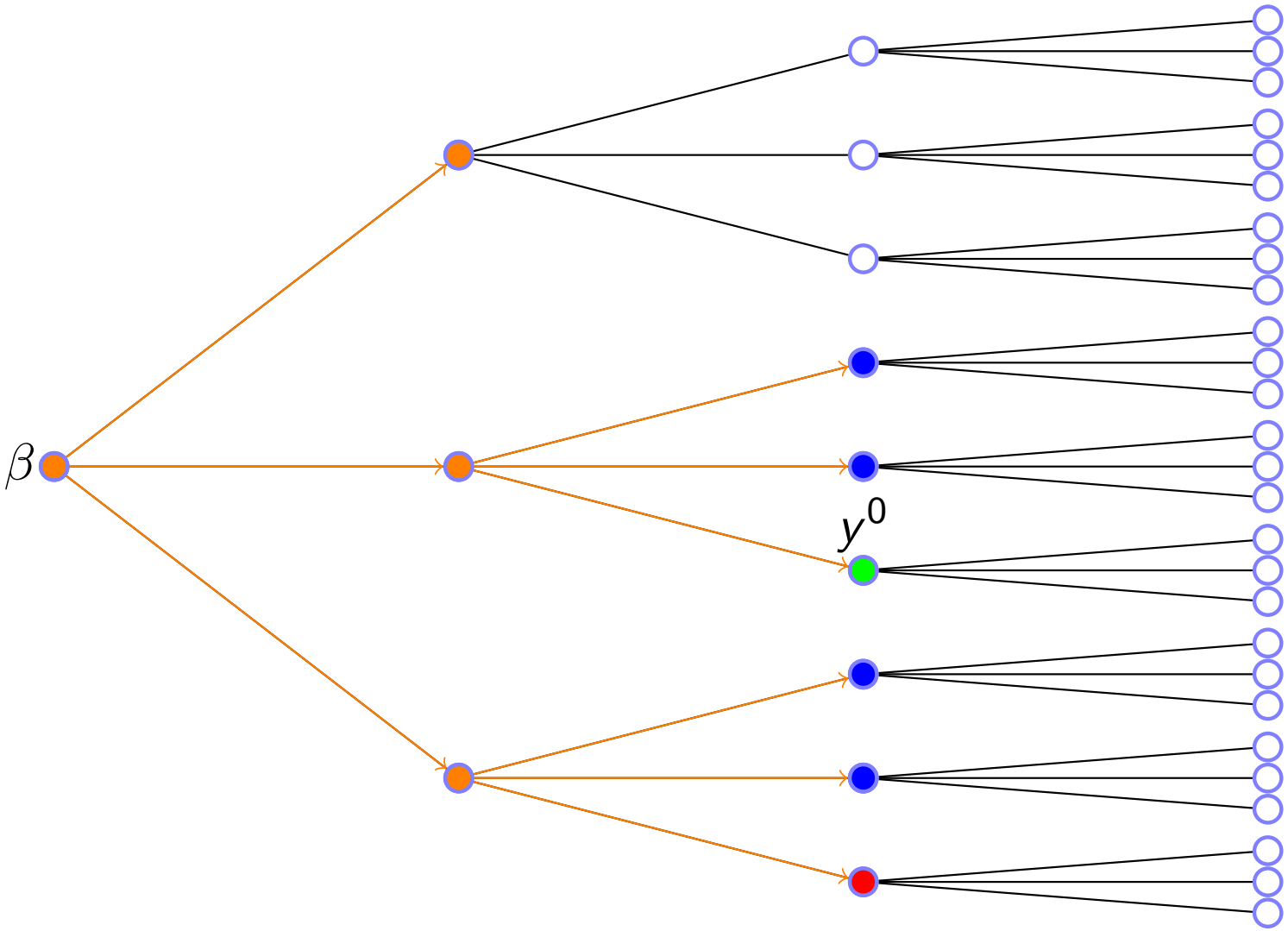

β
y^0

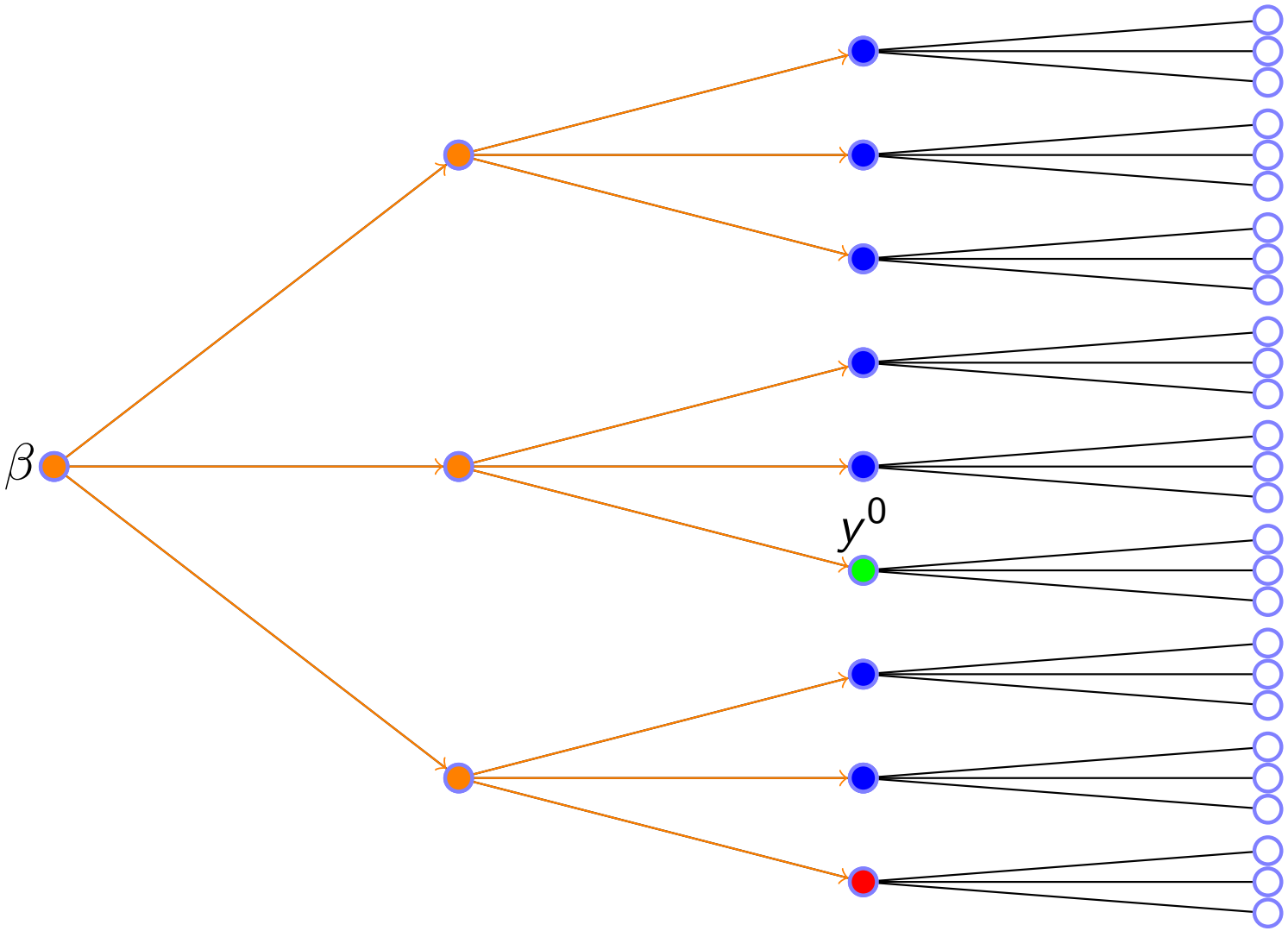

β
y^0


β
y^0


β
y^0


β
y^0


β
y^0


β
y^0

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{ij} \geq 1$$
$$0 \leq x_{i,j} \leq u_{i,j}$$
$$\sum_{ij} c_{ij} x_{ij} \leq c_G$$

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_j x_{ij} \geq 1 \\ & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

Para calcular $\mathcal{G}_{>_c}$, necesitamos expresar la región factible como $Ax = b$, así cada desigualdad, nos añade una variable de holgura.

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_j x_{ij} \geq 1 \\ & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

Para calcular $\mathcal{G}_{>_c}$, necesitamos expresar la región factible como $Ax = b$, así cada desigualdad, nos añade una variable de holgura.

$$\sum_j x_{ij} - d_i = 1$$

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_j x_{ij} \geq 1 \\ & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

Para calcular $\mathcal{G}_{>_c}$, necesitamos expresar la región factible como $Ax = b$, así cada desigualdad, nos añade una variable de holgura.

$$\begin{aligned} \sum_j x_{ij} - d_i &= 1 \\ x_{ij} + t_{ij} &= u_i \end{aligned}$$

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_j x_{ij} \geq 1 \\ & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

Para calcular $\mathcal{G}_{>_c}$, necesitamos expresar la región factible como $Ax = b$, así cada desigualdad, nos añade una variable de holgura.

$$\begin{aligned} \sum_j x_{ij} - d_i &= 1 \\ x_{ij} + t_{ij} &= u_i \\ \sum_{ij} c_{ij} x_{ij} + b &= c_G \end{aligned}$$

Calculo $\mathcal{G}_{>_c}$

$$(LP2) \quad \text{mín} \sum_{i,j} c_{ij} x_{ij}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_j x_{ij} \geq 1 \\ & 0 \leq x_{i,j} \leq u_{i,j} \\ & \sum_{ij} c_{ij} x_{ij} \leq c_G \end{aligned}$$

Para calcular $\mathcal{G}_{>_c}$, necesitamos expresar la región factible como $Ax = b$, así cada desigualdad, nos añade una variable de holgura.

$$\begin{aligned} \sum_j x_{ij} - d_i &= 1 \\ x_{ij} + t_{ij} &= u_i \\ \sum_{ij} c_{ij} x_{ij} + b &= c_G \end{aligned}$$

Supondremos que $c_{iq} \geq c_{ip}$ si $q < p$.

Usando las n primeras desigualdades: $\sum_j x_{ij} \geq 1$, llamamos:

$$D = \begin{pmatrix} \overbrace{1 \ldots 1}^{k_1} & \overbrace{0 \ldots 0}^{k_2} & \ldots & \overbrace{0 \ldots 0}^{k_n} \\ 0 \ldots 0 & 1 \ldots 1 & \ldots & 0 \ldots 0 \\ & & \ddots & \\ 0 \ldots 0 & 0 \ldots 0 & \ldots & 1 \ldots 1 \end{pmatrix}$$

Usando las n primeras desigualdades: $\sum_j x_{ij} \geq 1$, llamamos:

$$D = \begin{pmatrix} \overbrace{1\ldots1}^{k_1} & \overbrace{0\ldots0}^{k_2} & \ldots & \overbrace{0\ldots0}^{k_n} \\ 0\ldots0 & 1\ldots1 & \ldots & 0\ldots0 \\ & & \ddots & \\ 0\ldots0 & 0\ldots0 & \ldots & 1\ldots1 \end{pmatrix}$$

El sistema anterior viene dado por la matriz, por bloques:

$$\begin{pmatrix} D & -I_n & 0 & 0 \\ I_{k_1+\ldots k_n} & 0 & I_{k_1+\ldots k_n} & 0 \\ c_{11}\ldots c_{nk_n} & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{ij} \\ d_i \\ t_{ij} \\ b \end{pmatrix} = \begin{pmatrix} 1 \\ u_i \\ c_G \end{pmatrix}$$

Test set

- Si consideramos el ideal I_A asociado a la matriz anterior, la base de Gröbner reducida para un orden lexicográfico es:

Test set

- Si consideramos el ideal I_A asociado a la matriz anterior, la base de Gröbner reducida para un orden lexicográfico es:

$$\mathcal{G} = \{\underline{x_{ik}\, d_k} - t_{ik}\, b^{c_{ik}}, \underline{x_{iq}\, t_{ip}} - x_{ip}\, t_{iq}\, b^{c_{iq}-c_{ip}}\}$$

Test set

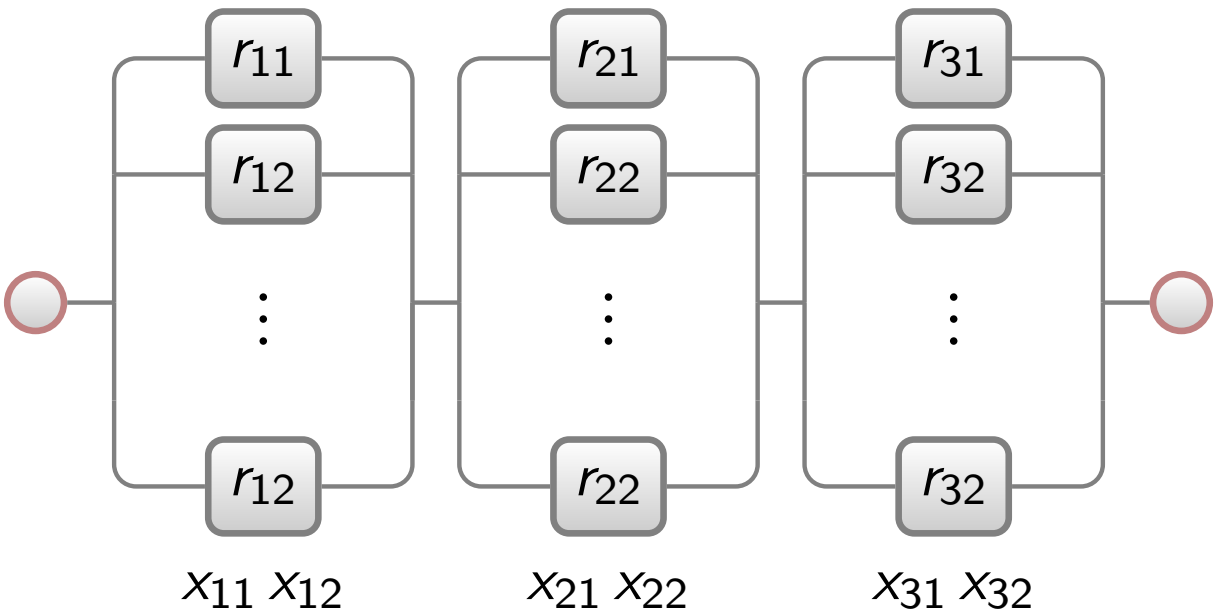
- Si consideramos el ideal I_A asociado a la matriz anterior, la base de Gröbner reducida para un orden lexicográfico es:

$$\mathcal{G} = \{\underline{x_{ik} d_k} - t_{ik} b^{c_{ik}}, \underline{x_{iq} t_{ip}} - x_{ip} t_{iq} b^{c_{iq} - c_{ip}}\}$$

- $\mathcal{G}$ también es base de Gröbner reducida para $>_c$.

Un ejemplo

Consideramos un sistema con 3 subsistemas en serie, y con 2 tipos de componentes para cada subsistema:



Un ejemplo

	r_{ij}	c_{ij}
$(1,1)$	0,9928	20
$(1,2)$	0,9901	15
$(2,1)$	0,9962	15
$(2,2)$	0,9948	11
$(3,1)$	0,9954	18
$(3,2)$	0,9931	15

Un ejemplo

	r_{ij}	c_{ij}
$(1,1)$	0,9928	20
$(1,2)$	0,9901	15
$(2,1)$	0,9962	15
$(2,2)$	0,9948	11
$(3,1)$	0,9954	18
$(3,2)$	0,9931	15

y	$R_0 = 0{,}99$	c
$y^0 = (0,1,0,2,0,2)$	0,99	67
$\beta = (0,1,0,1,0,1)$	0,9782	41
$(1,1,0,1,0,1)$	0,9879	61
$(0,2,0,1,0,1)$	0,9878	56
$(0,1,1,1,0,1)$	0,9832	56
$(0,1,0,2,0,1)$	0,9832	52
$(0,1,0,1,1,1)$	0,9849	59
$(0,1,0,1,0,2)$	0,9849	56
$(1,0,0,1,0,1)$	0,9808	46
$(0,1,1,0,0,1)$	0,9795	45
$(0,1,0,1,1,0)$	0,9804	44

Un ejemplo

y	$R_0 = 0{,}99$	c
$(1,1,0,1,1,0)$	0,9902	64
$(0,2,0,1,1,0)$	0,9901	59
$(0,1,1,1,1,0)$	0,9855	59
$(0,1,0,2,1,0)$	0,9855	55
$(1,0,0,1,1,0)$	0,9831	49
$(0,1,1,0,1,0)$	0,9818	48
$(1,0,1,0,0,1)$	0,9822	50
$(1,0,0,2,0,1)$	0,9859	57
$(1,0,1,0,1,0)$	0,9845	53

Un ejemplo

y	$R_0 = 0{,}99$	c
$(1,1,0,1,1,0)$	0,9902	64
$(0,2,0,1,1,0)$	0,9901	59
$(0,1,1,1,1,0)$	0,9855	59
$(0,1,0,2,1,0)$	0,9855	55
$(1,0,0,1,1,0)$	0,9831	49
$(0,1,1,0,1,0)$	0,9818	48
$(1,0,1,0,0,1)$	0,9822	50
$(1,0,0,2,0,1)$	0,9859	57
$(1,0,1,0,1,0)$	0,9845	53

Por tanto $(0,2,0,1,1,0)$ es el óptimo.

Tiempos Ruan & Sun

n	k	Nº medio de iteraciones	Nº medio de cajas	Tiempo medio de ejecución
10	2	696	1990	0,90
10	3	5797	15283	13,5
10	5	26427	65502	109,1
15	2	15184	36647	34,5
15	3	85103	188035	316,6
20	2	294747	907108	1031,9

Nuestros tiempos

n	k	Nodos	Tiempo	Ordenado
10	3	0	0,0	Sí
10	5	0	0,0	Sí
15	2	34	0,5	Sí
15	3	1655	34,5	Sí
17	2	5685	493,0	Sí
18	2	989	27,0	No
19	2	1899	64,0	No
20	2	5194	370,0	No

Nuestros tiempos

n	k		Nodos		Tiempo	Ordenado
10	3	5797	0	13,5	0,0	Sí
10	5	26427	0	109,1	0,0	Sí
15	2	15184	34	34,5	0,5	Sí
15	3	85103	1655	316,6	34,5	Sí
17	2		5685		493,0	Sí
18	2		989		27,0	No
19	2		1899		64,0	No
20	2	294747	5194	1031,9	370,0	No