Complejidad Temporal

Máquinas de Turing No Determinísticas (MTN)
(Para referirnos a las máquinas de Turing determinísticas utilizaremos MTD)

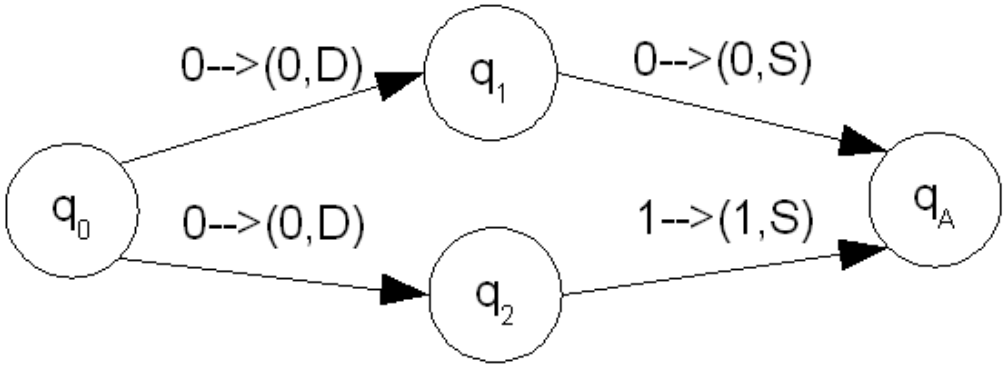
MTN es una Máquina de Turing que para un estado y un símbolo de la cinta barrido por el cabezal, tiene un conjunto finito de alternativas para el siguiente movimiento. La MTN acepta su entrada si cualquier sucesión de alternativas de movimiento se detiene en q_A (o en un estado final, en el otro modelo estudiado)

$$M=<Q, \Sigma, \Gamma, \Delta, q_0, q_A, q_R>, \quad \Delta: Q \times \Gamma \rightarrow \rho((Q \cup \{q_A, q_R\}) \times \Gamma \times \{D, I, S\})$$

M acepta su entrada *w*, sii existe al menos una computación a partir de *w* alcanza el estado q_A

Nota: Obsérvese que una MTD es un caso particular de una MTN donde el conjunto de alternativas es siempre un conjunto unitario

Ejemplo: Construir una MTN M tq L(M)={$w \in \{0,1,2,3\}^*$ / w comienza con 00 ó 01 }



En el gráfico faltan todas las otras combinaciones de estado símbolo que tienen que llevar al estado q_R.

$\Delta(q_0,0) = \{(q_1,0,D),(q_2,0,D)\}$
$\Delta(q_0,x) = \{(q_R,x,S)\}$ para todo $x \in \Gamma - \{0\}$
$\Delta(q_1,0) = \{(q_A,0,S)\}$
$\Delta(q_1,x) = \{(q_R,x,S)\}$ para todo $x \in \Gamma - \{0\}$
$\Delta(q_2,1) = \{(q_A,1,S)\}$
$\Delta(q_2,x) = \{(q_R,x,S)\}$ para todo $x \in \Gamma - \{1\}$

Traza de computación. Para una MTN la traza tendrá forma de árbol dónde cada rama representa una computación (una sucesión de alternativas de movimiento)

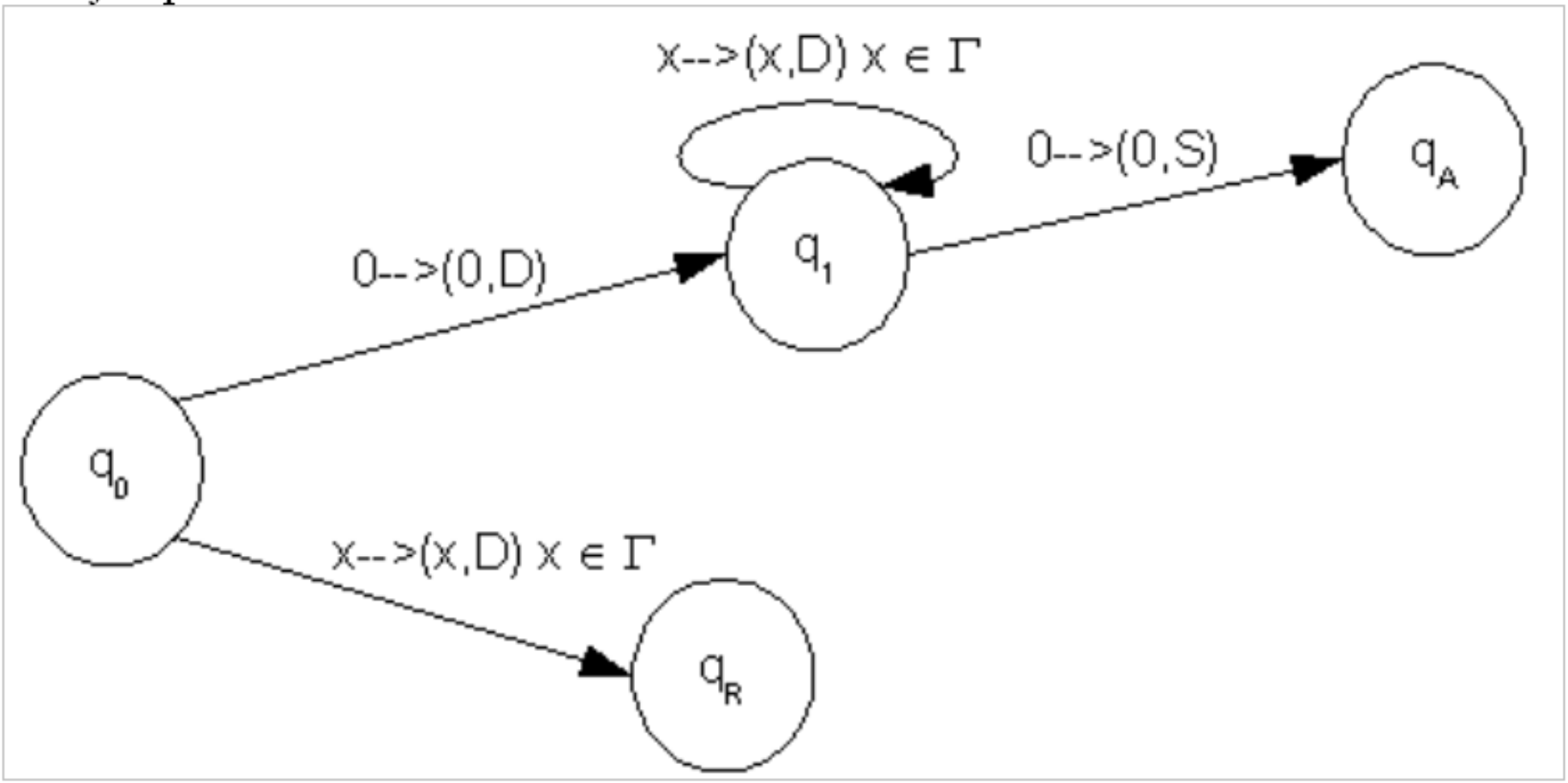
traza para w= 021

$\mathbf{q_0}021$

traza para w=0054

$\mathbf{q_0}0054$

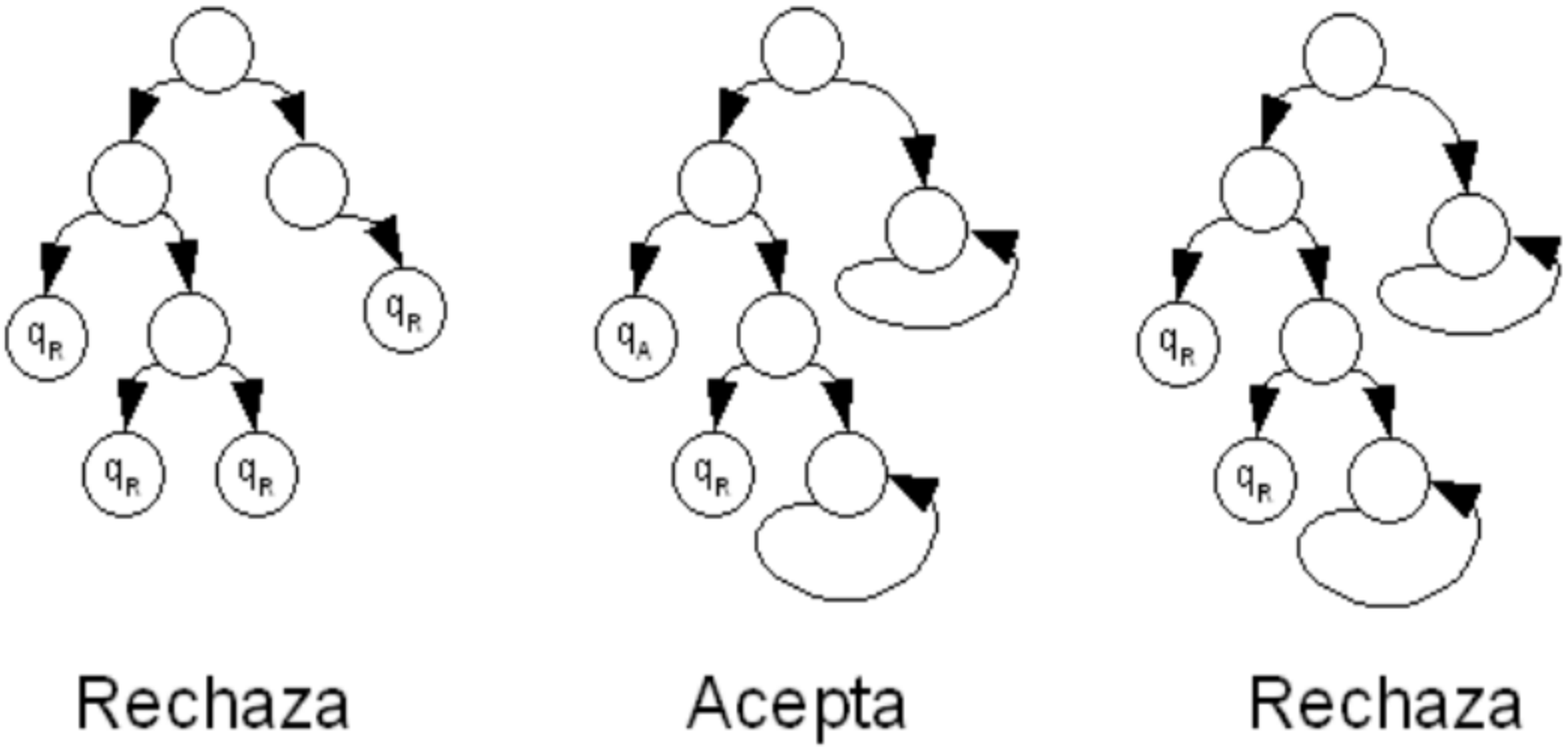
Otro ejemplo:



traza para $w = 0010$

$\mathbf{q_0}0010$

Gráfico de aceptación/rechazo

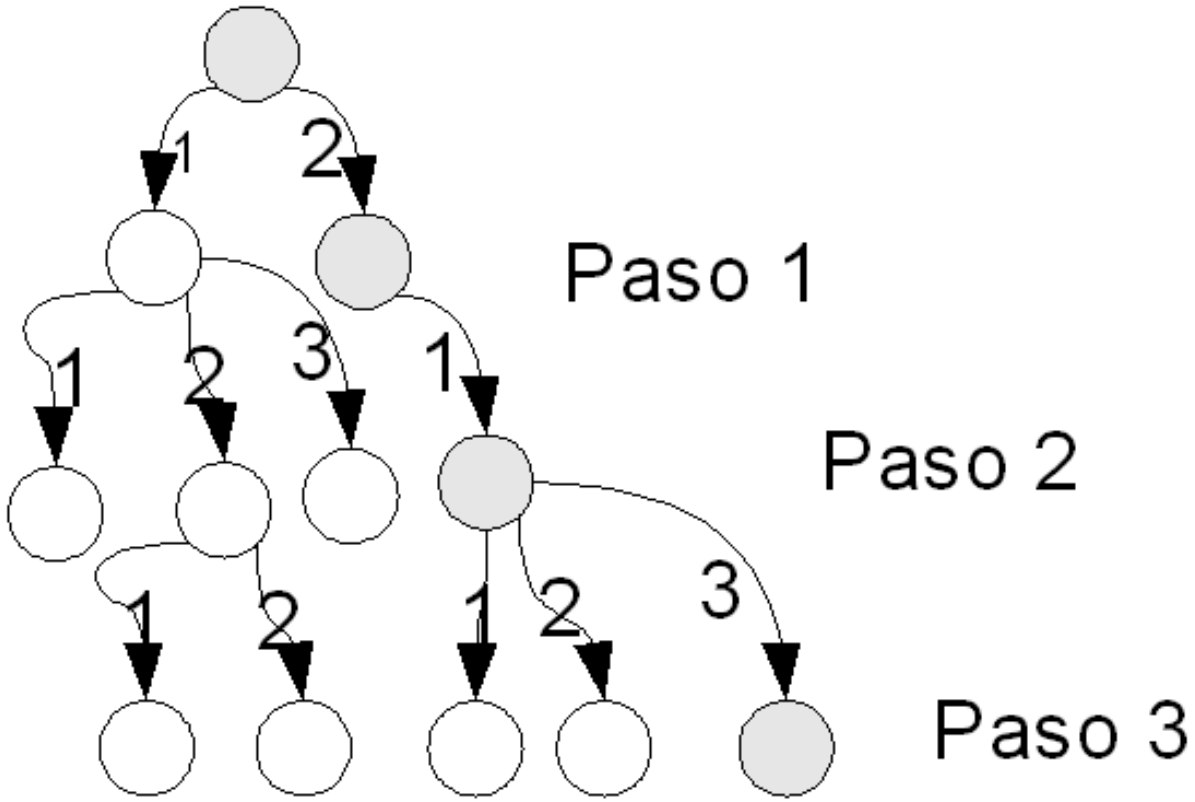


Ejercicio: Construir una MTN tq $L(M)=\{w \in \Sigma^* / cant_1(w) \ mod \ 2 = 0 \text{ or } cant_1(w) \ mod \ 3 = 0\}$

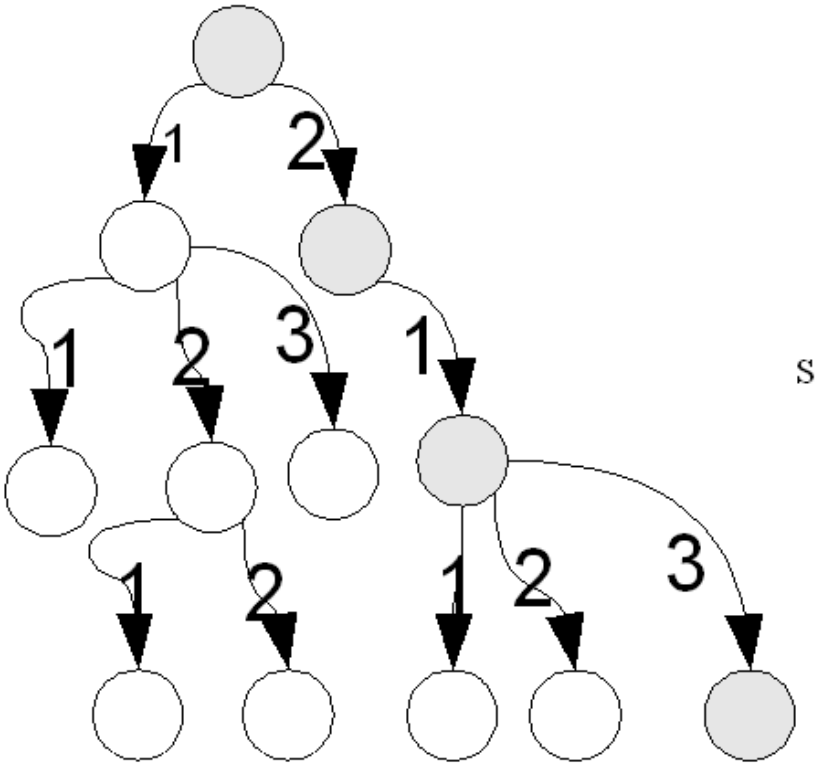
$\Sigma=\{0,1\}$

Teorema: Si L es un lenguaje aceptado por una MTN M_1 entonces L es aceptado por una MTD M_2.

Idea de la demostración: Realizar un recorrido BFS del árbol de computación.



Sea *r* el número máximo de alternativas para cualquier par (estado,símbolo), cualquier computación (rama del árbol) puede representarse por una secuencia de dígitos del 1 a *r*:



secuencia [2-1-3]

Construimos M_2 con 3 cintas. En la cinta 1 está la entrada, en la cinta 2 M_2 va generando a medida que lo necesite secuencias de dígitos de 1 a r en orden canónico (primero las más cortas, para igual longitud según orden numérico). Supongamos $r = 3$, se irá generando:

[1][2][3][1-1][1-2][1-3][2-1][2-2][2-3][3-1][3-2][3-3][1-1-1][1-1-2] …

Para cada secuencia que va generándose en la cinta 2 M_2 copia la entrada en la cinta 3 y simula M_1 sobre la cinta 3 utilizando la sencuencia de la cinta 2 para elegir los movimientos de M_1. Algunas secuencias se descartan pues no existen r alternativas para todos los casos. Si M_1 alcanza q_A, M_2 acepta parando en q_A.

Complejidad Temporal

Def. Sea M una MTD con k cintas, decimos que M es de complejidad t(n), si para toda entrada de longitud n, M hace a lo sumo t(n) movimientos.

Def. Sea M una MTN con k cintas, decimos que M es de complejidad t(n), si ninguna secuencia de alternativas de movimiento ocasiona que M haga más de t(n) movimientos.

Clases de complejidad

Def. Un lenguaje $L \in DTIME(t(n))$ sii existe una MTD M, con $L=L(M)$ tal que la complejidad temporal de M pertenece a $O(t(n))$

Def. Un lenguaje $L \in NTIME(t(n))$ sii existe una MTN M, con $L=L(M)$ tal que la complejidad temporal de M pertenece a $O(t(n))$

Corolario: $DTIME(t(n))$ está incuido en $NTIME(t(n))$

Def. La clase de lenguajes P comprende todos los lenguajes para los que existe una MTD que lo acepta en tiempo polinómico

$$P=U_{i\geq 0}\ DTIME(n^i)$$

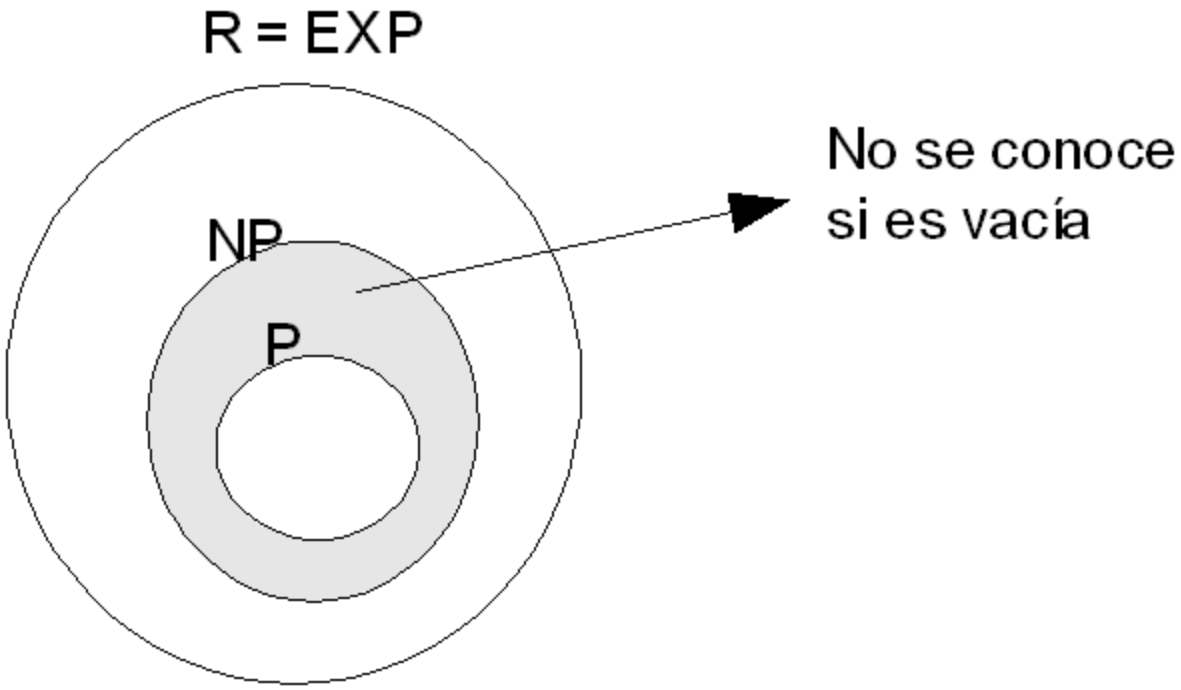
Def. La clase de lenguajes NP comprende todos los lenguajes para los que existe una MTN que lo acepta en tiempo polinómico

$$NP=U_{i\geq 0}\ NTIME(n^i)$$

Corolario: P es un subconjunto de NP (inmediato, pues las MTD son un caso particular de las MTN)

Teorema: Si L es un lenguaje recursivo aceptado por una MTN M_1 en tiempo t(n), existe una MTD M_2 que lo acepta en tiempo menor o igual que $d^{t(n)}$, con d una constante mayor que 1 que depende de M_1

NOTA: Se sabeque existen lenguajes recursivos que no pertenecen a NP ($R\text{-}NP \neq \emptyset$) pero no se sabe si NP=P

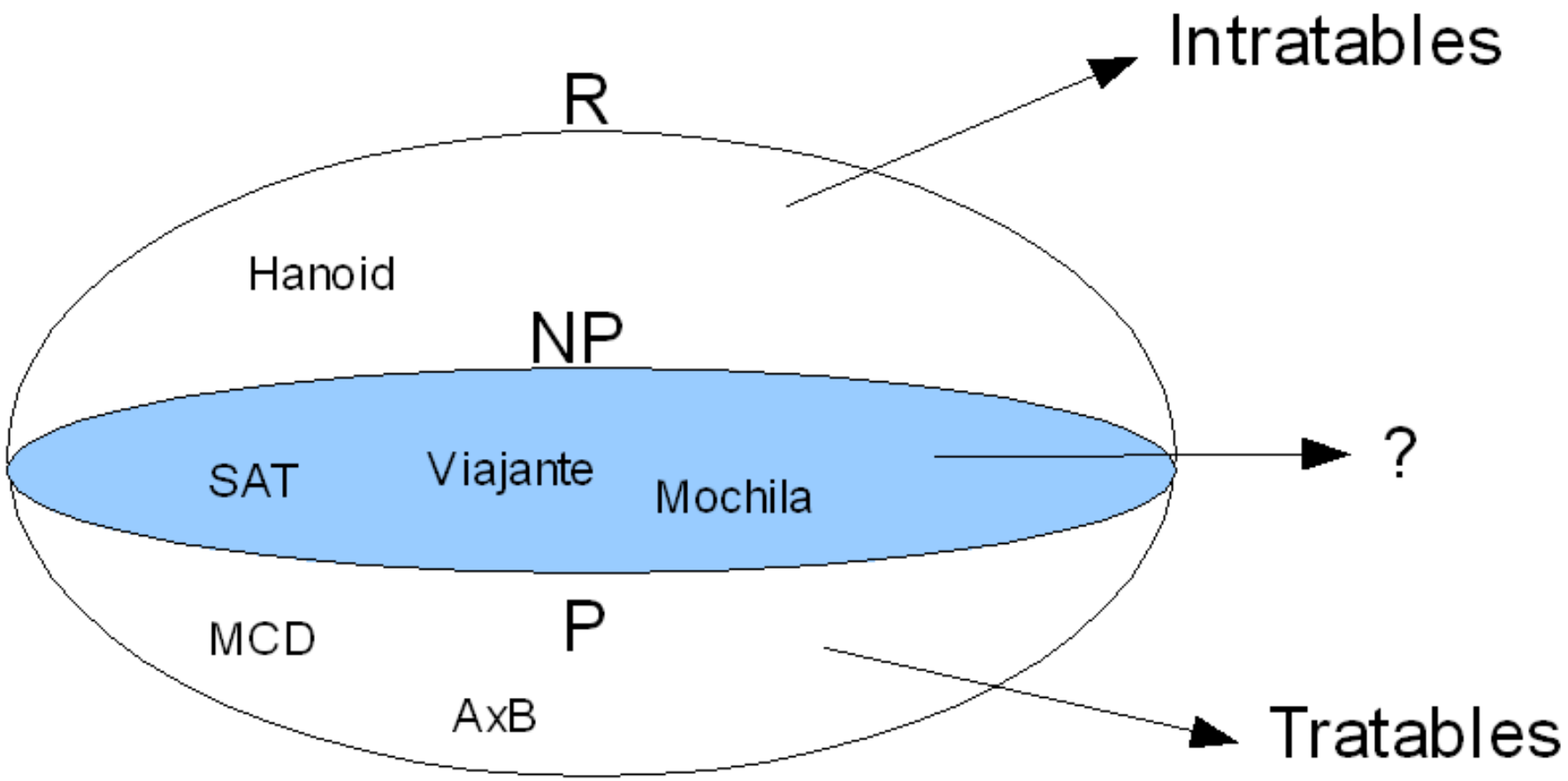


Tratabilidad. En general se está de acuerdo que la clase P representa a los problemas tratables. La mayoría de los problemas de esta clase tienen algoritmos con implementación razonable (Máximo Común Divisor, multiplicación de matrices, 2-SAT, etc).

Sin embargo existen ciertos problemas en P que realmente no son tratables debido a:

- el grado del polinomio es muy grande
- las ctes ocultas de la notación O son muy grandes
- la demostración de pertenencia a P no es constructiva y no se conoce algoritmo eficiente

Pero estos son casos extremos, y se considera a P una razonable aproximación al concepto de tratabilidad



Esta brecha es inaceptablemente muy grande pues existen muchos problemas importantes que pertenecen a NP a los que no se le concoce una solución tratable, pero tampoco se ha demostrado que esta solución no exista.