Scheduling
o planificación de observaciones

Francisco Förster Burón

Astroinformática – 18 Oct 2011

Motivación:

¿Cómo planificar observaciones astronómicas automáticamente en un telescopio robótico?

Las condiciones atmosféricas pueden variar sin previo aviso.

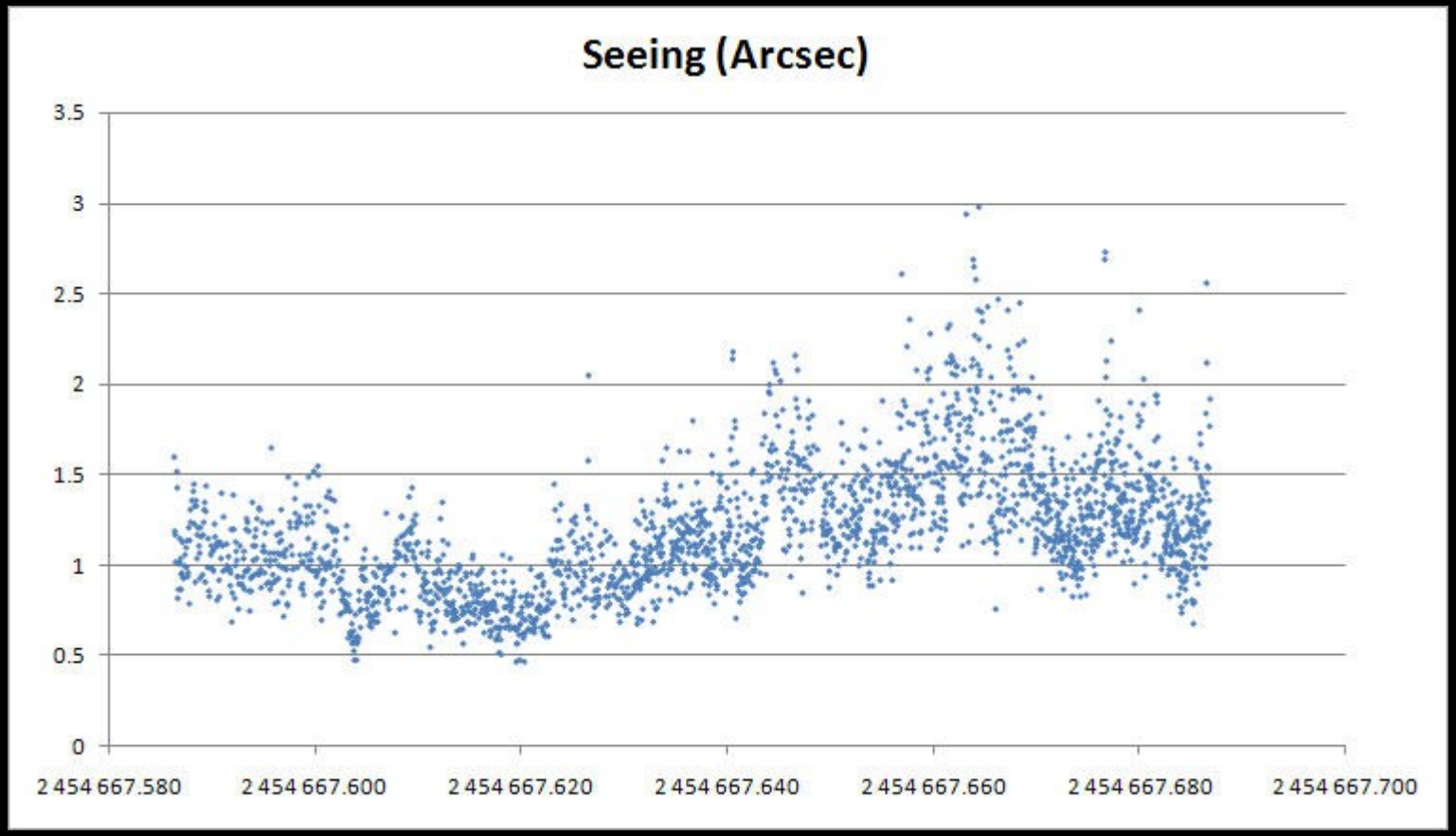
La rotación de la Tierra hace que la posición de los objetos varíe en el cielo.

Es posible que se quieran satisfacer varios objetivos científicos (e.g. varios investigadores)

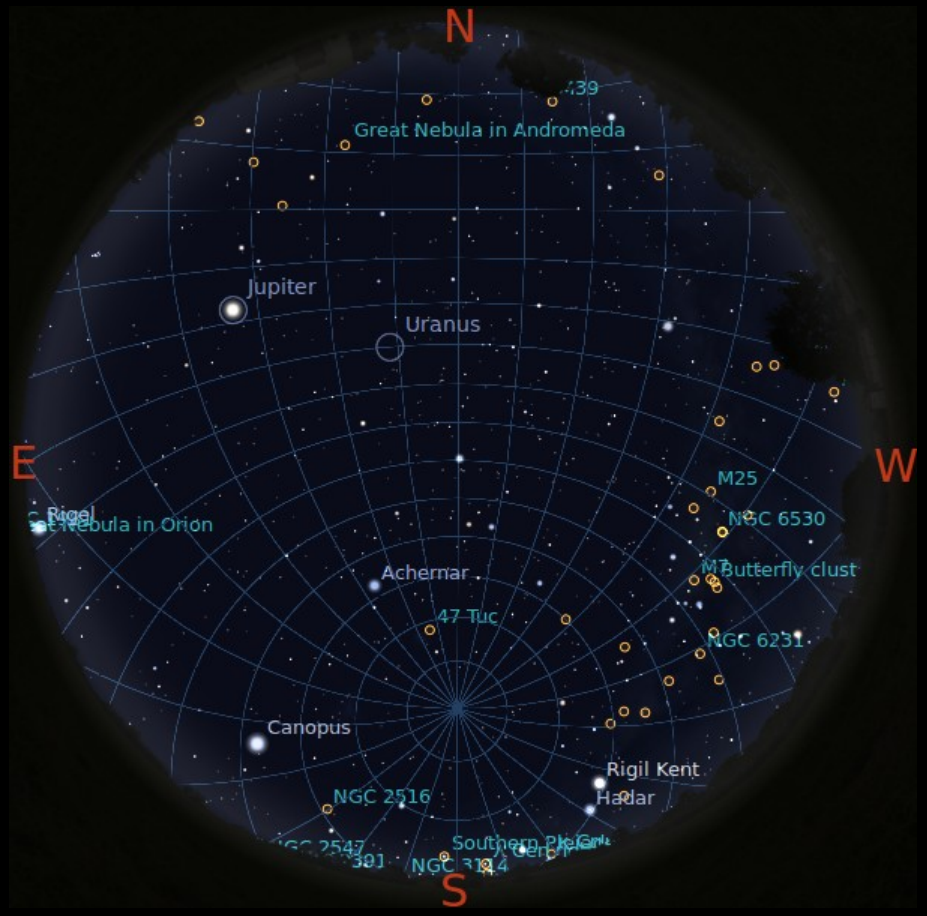
Se debe considerar el tiempo perdido moviendo el telescopio entre objetos.

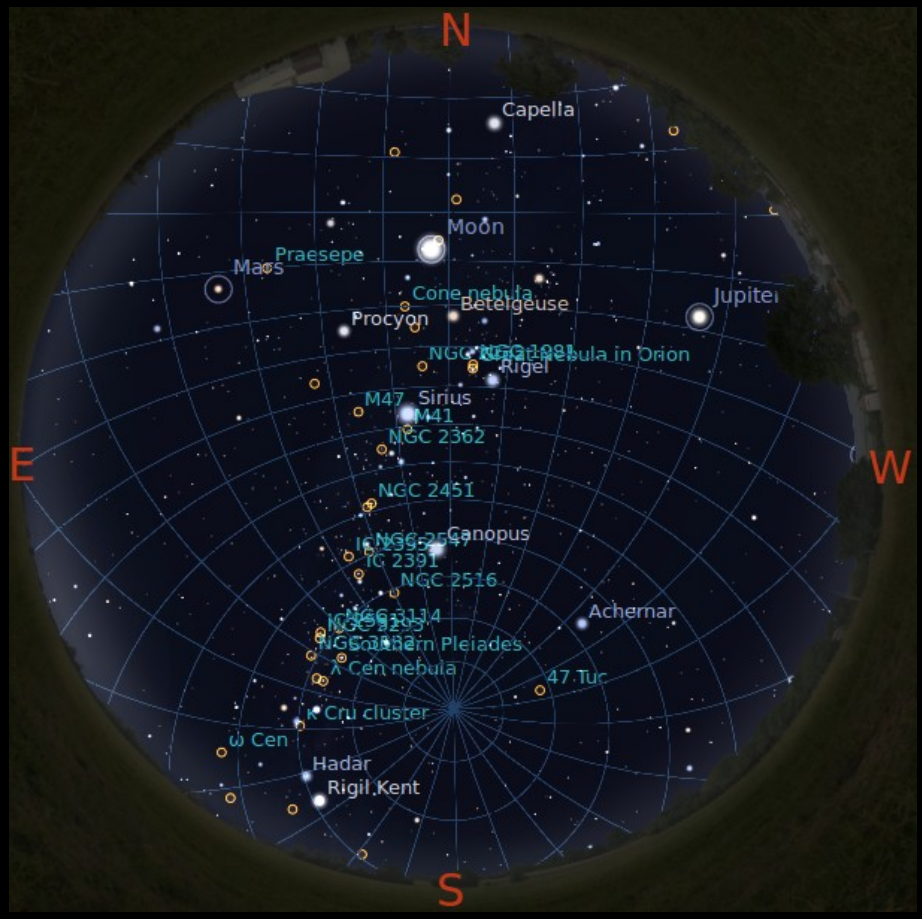
La utilidad científica de cada observación puede depender del momento en que se haga

Optimización tradicional no sirve dada la complejidad del problema, se necesitan *métodos heurísticos*



Ejemplo de variaciones del seeing en La Silla a lo largo de una noche (Mar 2005)

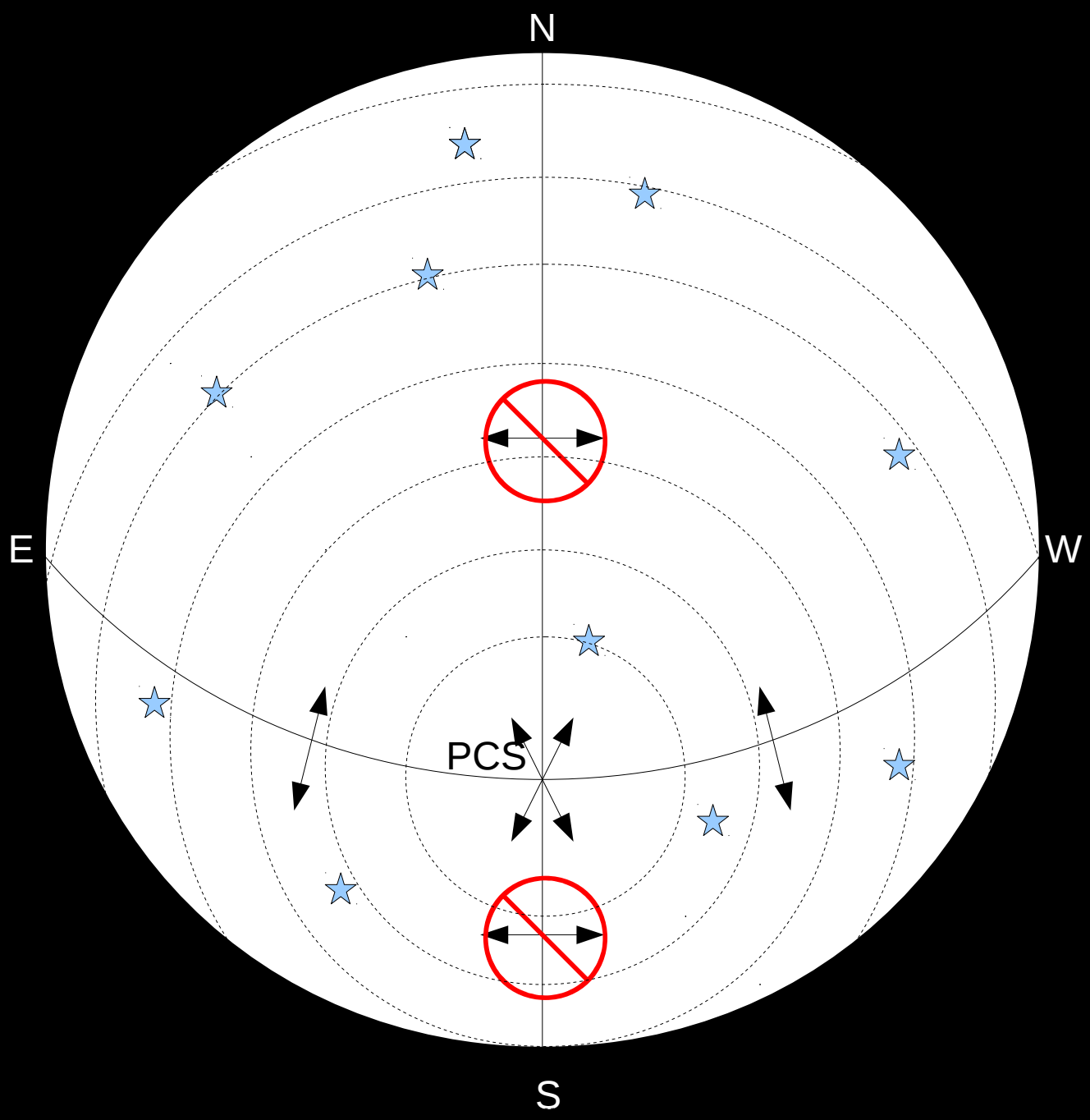


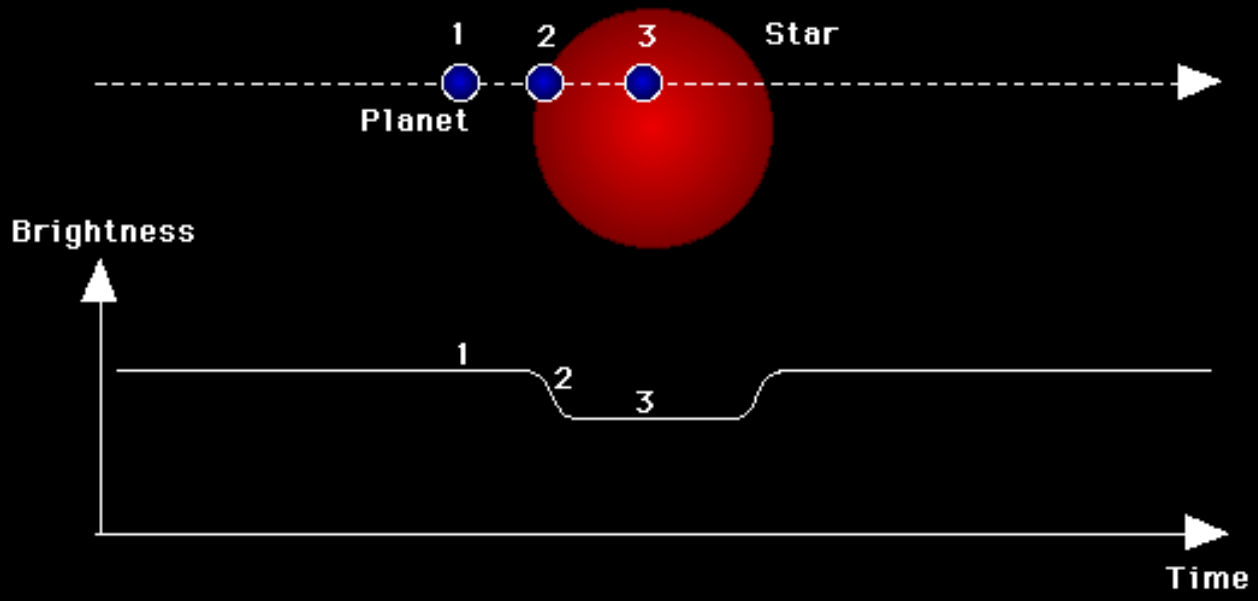


Cambio de la posición de las estrellas y del brillo del cielo debido a la rotación de la Tierra.
Círculo de ~50° de radio alrededor de la Luna debe evitarse para no contaminar las observaciones.

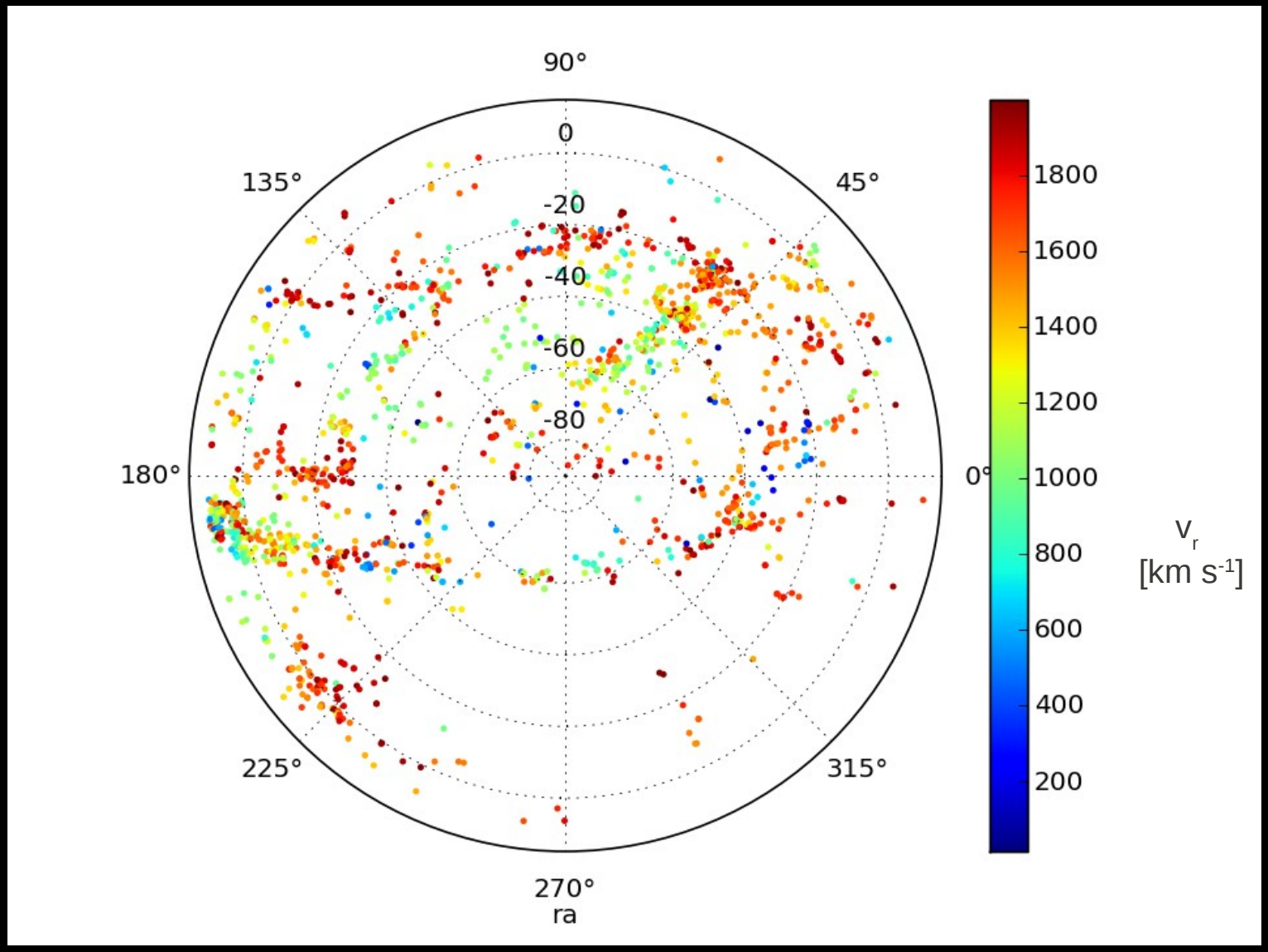
Movimientos permitidos en una montura ecuatorial

Es conveniente cambiar de meridiano sólo en observaciones cercanas al polo, o cambiar de meridiano muy pocas veces a lo largo de la noche.





Algunos eventos son visibles en ventanas de tiempo muy pequeñas.
Si una observación del programa se atrasa es posible arruinar las observaciones posteriores.



Número de objetos disponibles para visitar puede ser muy grande.
Ejemplo: galaxias más cercanas del programa de búsqueda de supernovas CHASE.

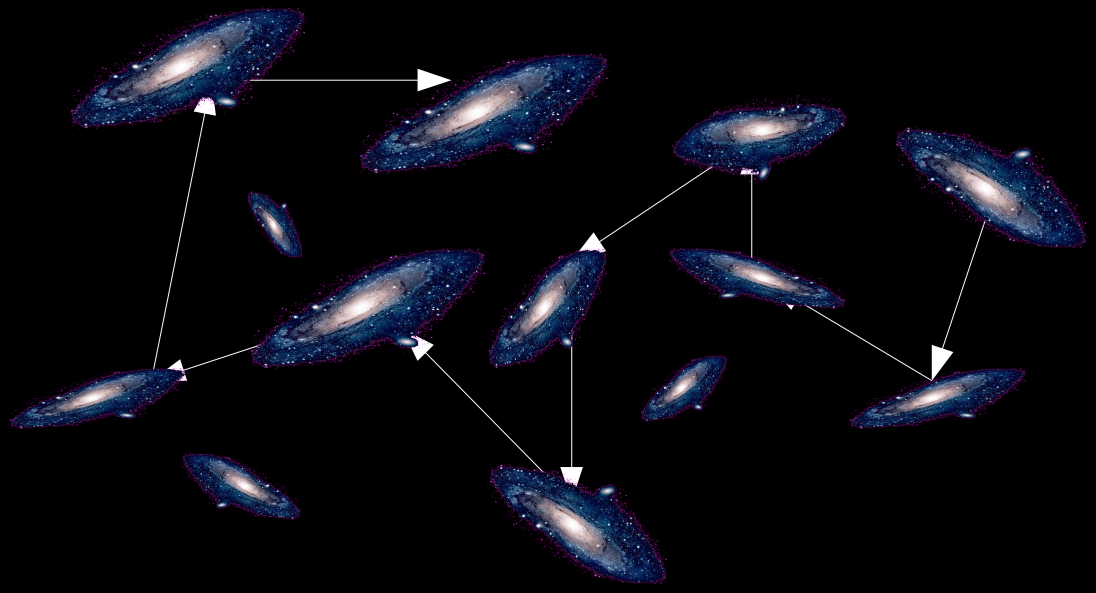
Número de variables a considerar implica número de combinaciones muy grande.

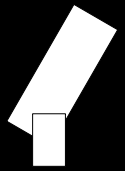
Búsqueda ingenua de todas las combinaciones posibles excede la capacidad de los computadores actuales.
Métodos tradicionales de optimización sufren de un exceso de mínimos locales.

Caso de estudio:
Detección de objetos variables en una muestra de galaxias

Primero discutiremos una versión simplificada del problema anterior: maximizar la tasa de detección de objetos variables en un levantamiento robótico de galaxias.

Lo primero que debemos definir es una medida de la calidad de un plan de observación. En nuestro caso el objetivo final es encontrar supernovas jóvenes para ser estudiadas con telescopios de 8 m





Telescopio robótico

Galaxias:

~1 supernova cada 100 años aproximadamente

x 10 de variación en la tasa de explosión

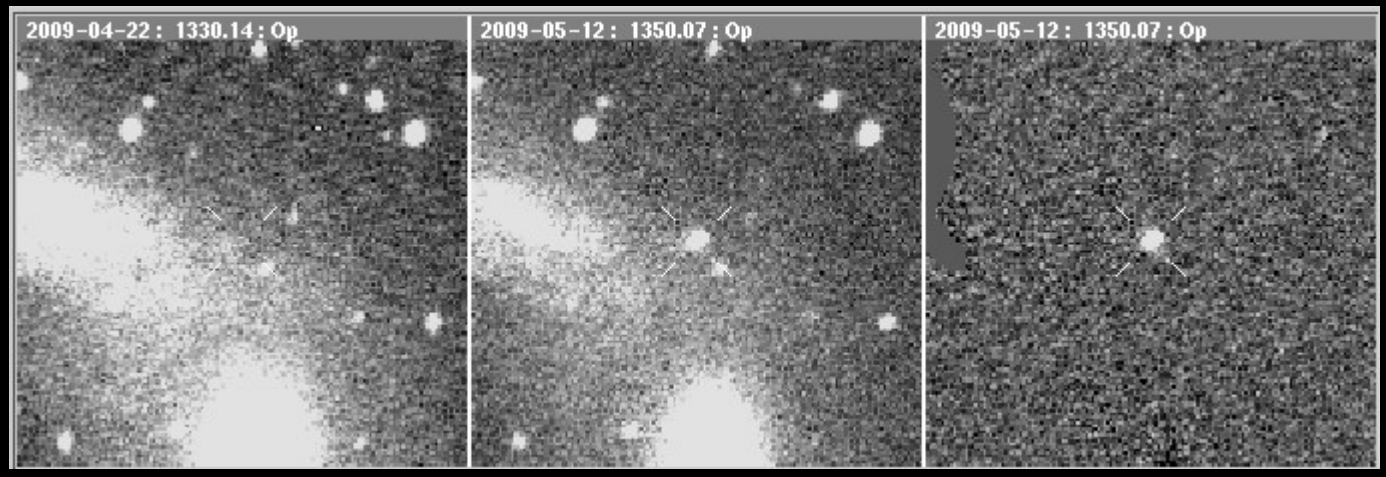
~10,000 galaxias en el Universo cercano donde es posible detectar supernovas con un telescopio robótico

Supernovas:

~1 a 10 días para hacerse visibles luego de la explosión con un telescopio robótico pequeño

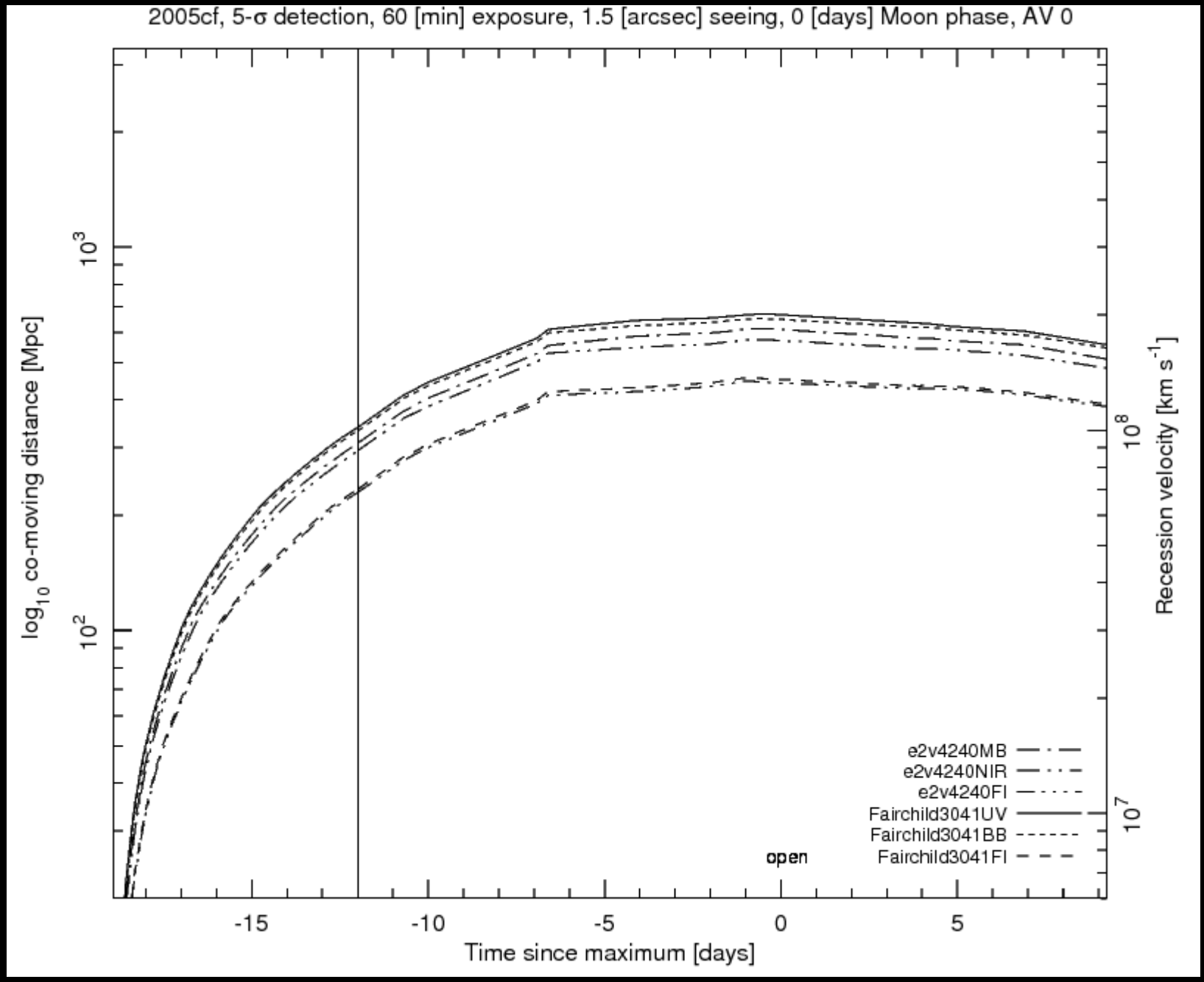
~20 días para alcanzar el máximo de la curva de luz luego de la explosión

Visibles por semanas, meses o años luego de la explosión dependiendo del tipo de supernova.



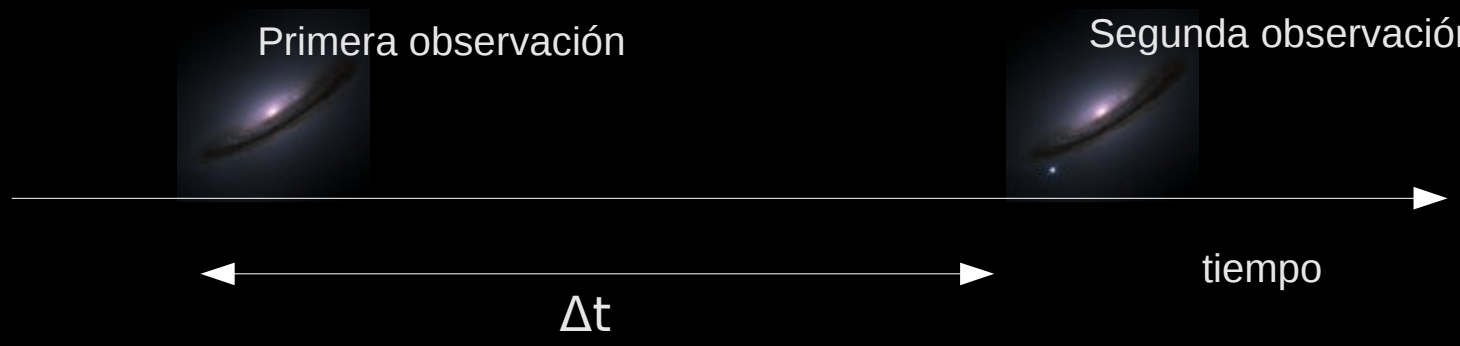
Ejemplo de supernova encontrada por CHASE

Distancia vs tiempo de detección



Número esperado de eventos 1

Las observaciones se repiten con una cadencia Δt



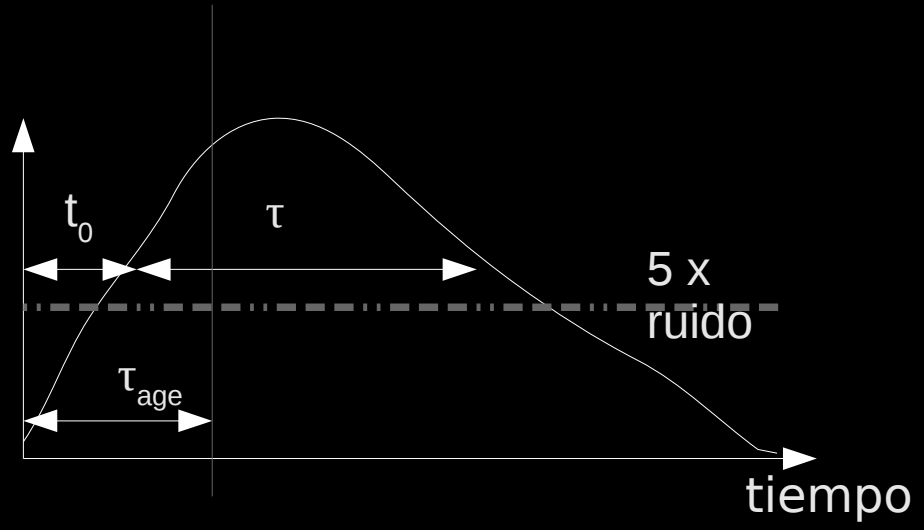
Supuestos:

La tasa de explosión de supernovas es R

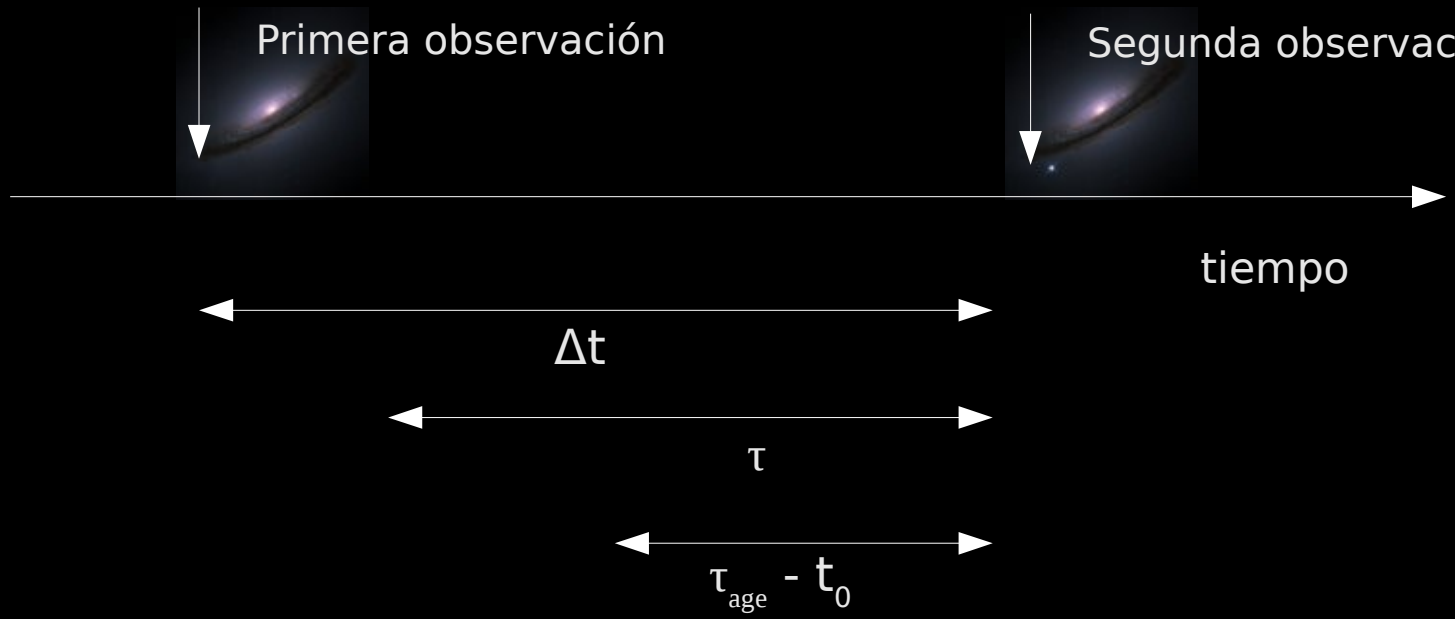
Demoran un tiempo t_0 en ser detectables.

Son visibles por un tiempo τ

Queremos encontrar eventos más jóvenes que aquellos con edad τ_{age}



Número esperado de eventos 2

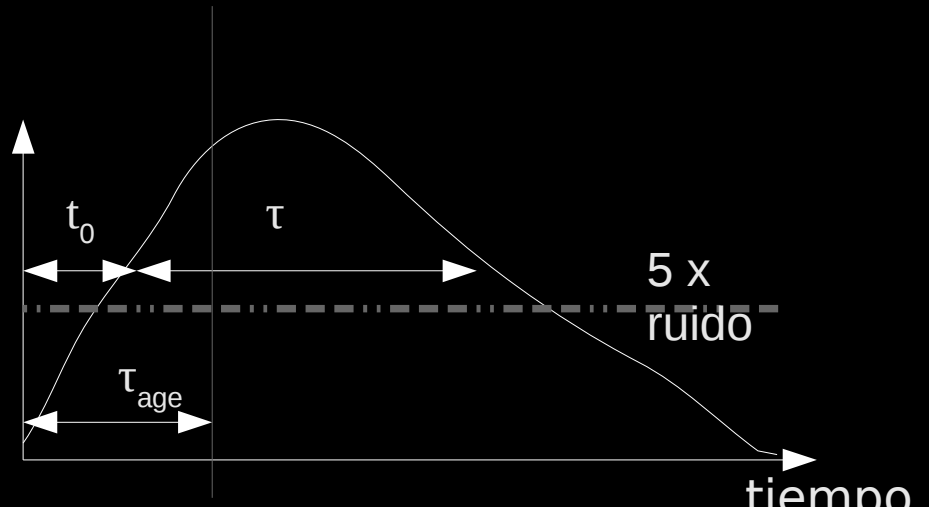


$$\lambda = R \cdot max\left[0,\ min\left(\tau_{age} - t_0,\ \tau,\ \Delta t\right)\right]$$

λ es el número esperado de eventos

$$P(k,\ \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Probabilidad de detectar k eventos en la segunda observación.



Número esperado de eventos 3

$$P(0, \lambda_i) = e^{-\lambda_i}$$

Probablidad de cero eventos detectados en la observación *i*

$$P(0, \{\lambda_i\}) = \prod_i e^{-\lambda_i} = e^{-\sum_i \lambda_i}$$

Probablidad de cero eventos detectados durante todo el plan de observaciones

Es decir, buscar el mejor plan de observaciones equivale a maximizar la sumatoria del número de eventos esperados durante todo el plan de observación.

Pero para calcular λ_i necesitamos calcular t_0 y τ para cada observación, lo que puede variar a medida que el objeto se mueve en el cielo.

Cálculo de t_0 y τ

1. Ecuación de señal a ruido

Razón señal a ruido

Electrones por unidad de tiempo debido al objeto variable

Tiempo de exposición

$$S/N(t) = \frac{\gamma_{\rm TE}^{\rm CCD}(t)\ T}{\left[\gamma_{\rm TE}^{\rm CCD}(t)\ T + \gamma_{\rm sky}^{\rm CCD}\ T\ n_{\rm pix} + \gamma_{\rm RN}^{2}\ n_{\rm pix}\right]^{1/2}}$$

Electrones por unidad de tiempo debido al brillo del cielo (+ electrones corriente oscura + electrones brillo galaxia)

Número de pixeles en una fuente puntual

Ruido de lectura del CCD

Cálculo de t_0 y τ

2. Escalando con la distancia y el ángulo zenital

Tasa de fotones incidentes provenientes de la fuente a una distancia arbitraria *D* dado que conocemos la tasa de fotones que se obtendrían a una distancia conocida D_0

$$\gamma^{\mathrm{CCD}}_{\mathrm{TE},D}(t) = \gamma^{\mathrm{CCD}}_{\mathrm{TE},D_0}(t) \left(\frac{D_0}{D}\right)^2$$

$$\gamma^{\mathrm{CCD}}_{\mathrm{SKY}}(z) = \frac{\gamma^{\mathrm{CCD}}_{\mathrm{SKY}}(0)}{\cos z}$$

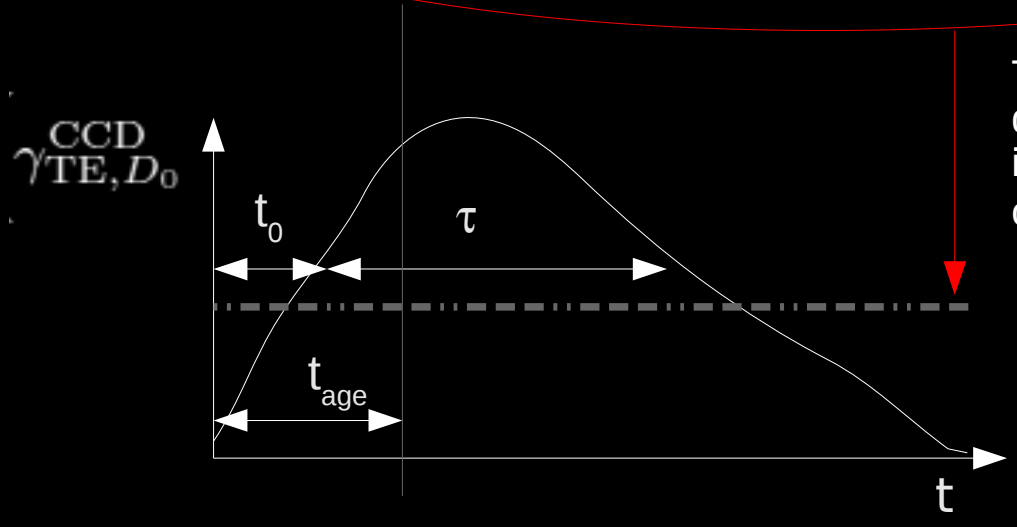
Tasa de fotones incidentes provenientes del cielo en la dirección de la fuente para un ángulo zenital arbitrario, dado que conocemos el número de fotones que se obtendrían mirando al zenith

Cálculo de t_0 y τ

3. Solución

Dos soluciones para una función que contiene una componente monotónicamente creciente y otra monotónicamente decreciente.

$$t_0,\ t_0+\tau = \left(\gamma^{\rm CCD}_{{\rm TE},D_0}\right)^{-1}\left\{\left(\frac{D}{D_0}\right)^2\frac{S/N^2}{2\,T}\left[1+\left(1+\frac{4\,n_{\rm pix}}{S/N^2}\{\gamma^2_{\rm RN}+\frac{\gamma^{\rm CCD}_{\rm sky}(0)}{\cos z}\,T\}\right)^{1/2}\right]\right\}$$



Tasa de fotones necesaria para la detección usando **una sola función a invertir**, en vez de una función por objeto, simplificando el cálculo de λ_i's

Cálculo de t_0 y τ

4. Solución basada en el pixel central

$$t_0,\, t_0+\tau = \left(\gamma_{\mathrm{TE},D_0}^{\mathrm{CCD}}\right)^{-1}\left\{\left(\frac{D}{D_0}\right)^2 \frac{S/N^2}{2\,f\,T}\left[1+\left(1+\frac{4}{S/N^2}\{\gamma_{\mathrm{RN}}^2+\frac{\gamma_{\mathrm{sky}}^{\mathrm{CCD}}(0)}{\cos z}\,T\}\right)^{1/2}\right]\right\}$$

Fracción de la luz que cae en el pixel central de la fuente a ser detectada, aproximado por la siguiente ecuación:

$$f = 1-\exp\left(-0.883\cdot\cos(z)^{1.2}\cdot\frac{Area_{pix}}{FWHM^2}\right)$$

Elección de la cadencia?

$$\lambda = R \cdot max\left[0,\ min\left(\tau_{age} - t_0,\ \tau,\ \Delta t\right)\right]$$

Para maximizar la fórmula superior se puede elegir:

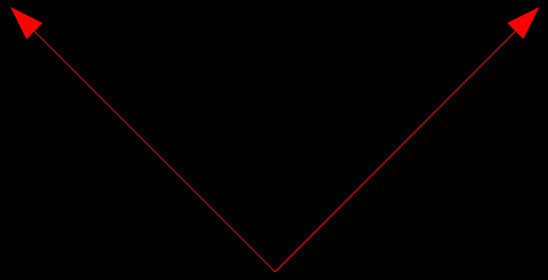
$$\Delta t = min\left(\tau_{age} - t_0,\ \tau\right)$$

Para eventos muy jóvenes en galaxias cercanas se tendrá:

$$\Delta t = \tau_{age} - t_0$$

Estadísticas asumiendo cadencia fija

Age	Reference Cadence	Sample size	Approx. detection rate
$\tau_{\rm age}$	$\Delta t = \langle \tau_{\rm age} - t_0 \rangle$	$N_{\rm exp}\ \Delta t$	$N_{\rm exp}\ \Delta t\ \langle R \rangle$



N_{exp}: # observaciones por noche

Sin embargo, es difícil mantener las cadencias fijas para cada objeto, debido a cambios no esperados de las condiciones de observación o cambios de prioridades.

Además, t_0 y τ varían con las condiciones atmosféricas y con la altura sobre el horizonte.

En lugar de intentar controlar la cadencia, el programa de optimización puede elegir la cadencia por nosotros, lo que está implicito en el cálculo de λ_i para cada observación

Fitness (Aptitud)

La figura de mérito que usaremos, también llamada *fitness* o aptitud será la siguiente:

$$\lambda^{\mathrm{age}}_{\mathrm{D},\ \mathrm{Total}} = \Sigma_i\ R^i\ \ \min\{\Delta t^i,\ \max(\tau_{\mathrm{age}} - t^i_0,\ 0),\ \tau^i\}$$

Conclusiones caso de estudio

A pesar de la complejidad del problema, es posible cuantificar la aptitud de cada plan de observaciones si hay claridad sobre los objetivos científicos.

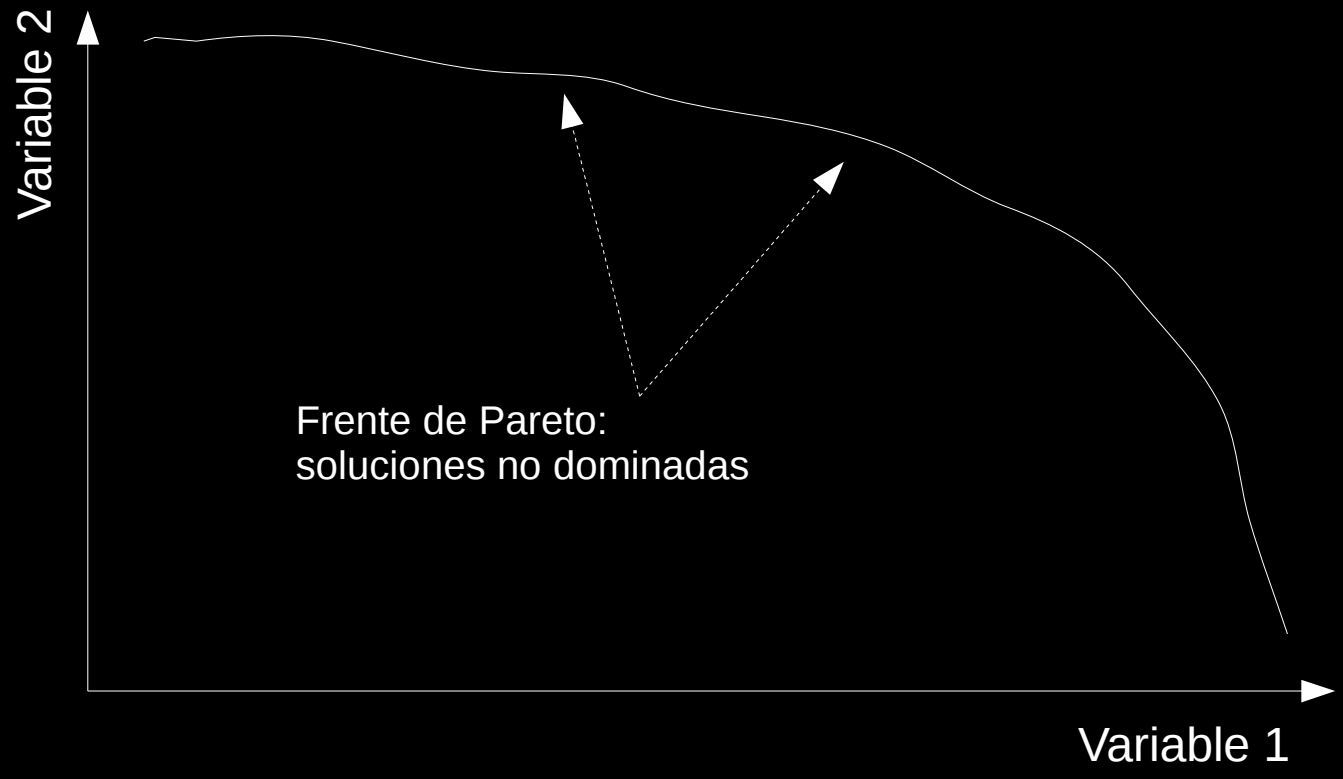
Es posible comparar planes de observación de forma objetiva usando la ecuación de señal a ruido y un poco de probabilidades.

Necesitamos una herramienta para elegir un plan entre la enorme variedad de planes posibles

Frente de Pareto
(optimización multiobjetivo)

Si bien para optimizar la aptitud usando una sola función objetivo es claro cómo comparar soluciones, para optimizar la aptitud con varios objetivos simultáneos se debe buscar el espacio de soluciones que maximiza los objetivos individuales sin afectar otros objetivos, esto se conoce como el **frente de Pareto.**

El problema entonces se reduce a elegir una solución particular del frente de Pareto por medio de algún criterio arbitrario.



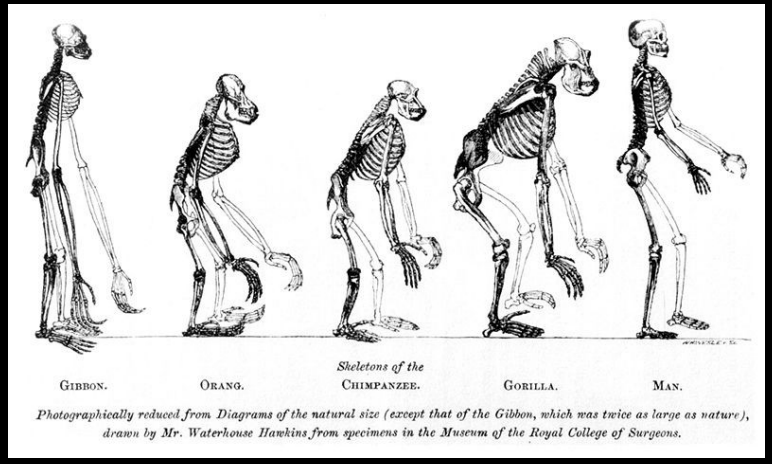
Métodos de optimización en problemas complejos

Para resolver este tipo de problemas es posible buscar inspiración en la naturaleza, que utiliza métodos de optimización muy exitosos que se basan en reglas muy simples.

Métodos basados en reglas simples y sin una justificación matemática rigurosa son conocidos como ***métodos heurísticos*.**

Los métodos heurísticos más populares están inspirados en sistemas biológicos, e.g. algoritmos genéticos, algoritmos de inteligencia de enjambre o redes neuronales.

En particular discutiremos dos familias de métodos: **algoritmos genéticos** y métodos inspirados en colonias de hormigas (***ant colony optimization***)



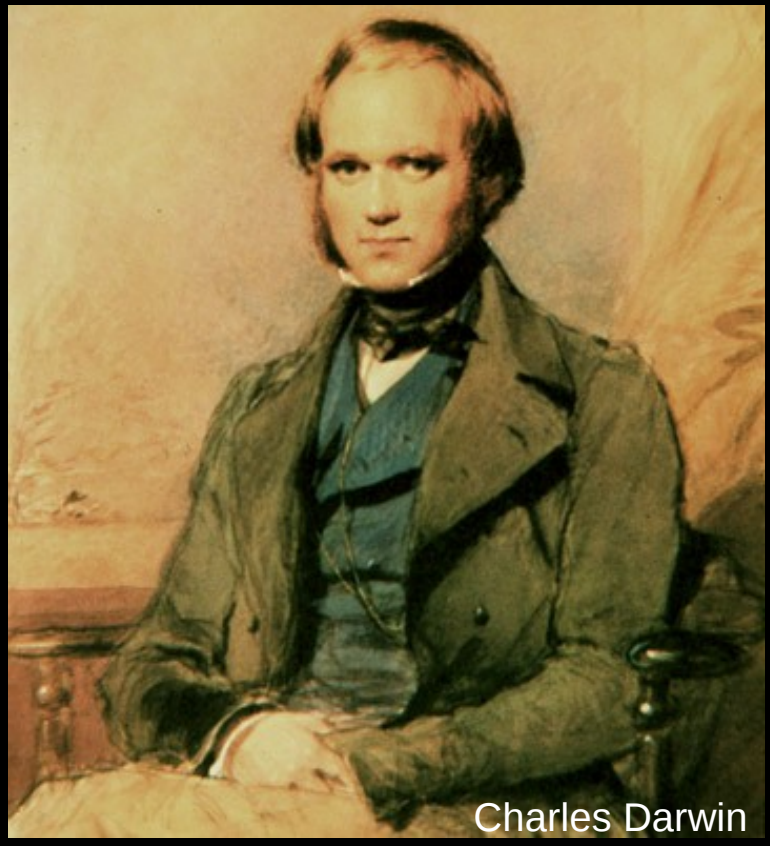
I. Algoritmos genéticos

Inspirado por la evolución las especies en la naturaleza que utiliza la prueba y error como herramienta básica.

La aptitud de los individuos para cierto ambiente está definida por qué tan bien ellos se adaptan para conseguir su objetivo: sobrevivir y multiplicarse.

En el caso computacional también se utiliza la prueba y error.

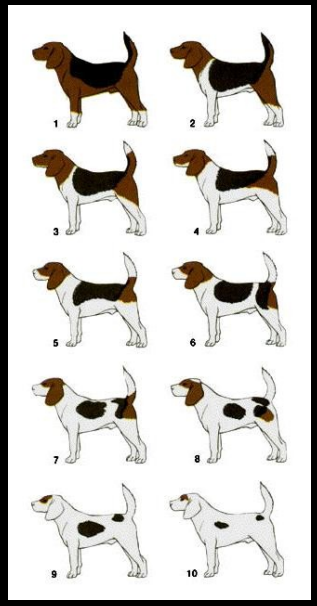
A partir de una colección de soluciones candidatas, su aptitud para satisfacer ciertos objetivos determina la probabilidad con que ellas serán mantenidas y usadas para construir más soluciones candidatas.

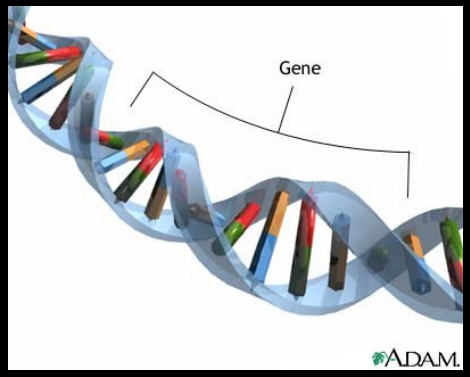

Charles Darwin

Selección natural: fenotipo y genotipo

Fenotipo: rasgos físicos y conductuales observables de un individuo, son la expresión del genotipo. Es la propiedad que la naturaleza evalúa al operar la selección natural.

En el caso de estudio correspondería a la señal a ruido de cada observación, el número de objetos observados y otras variables que son evaluadas como un número de supernovas esperadas.





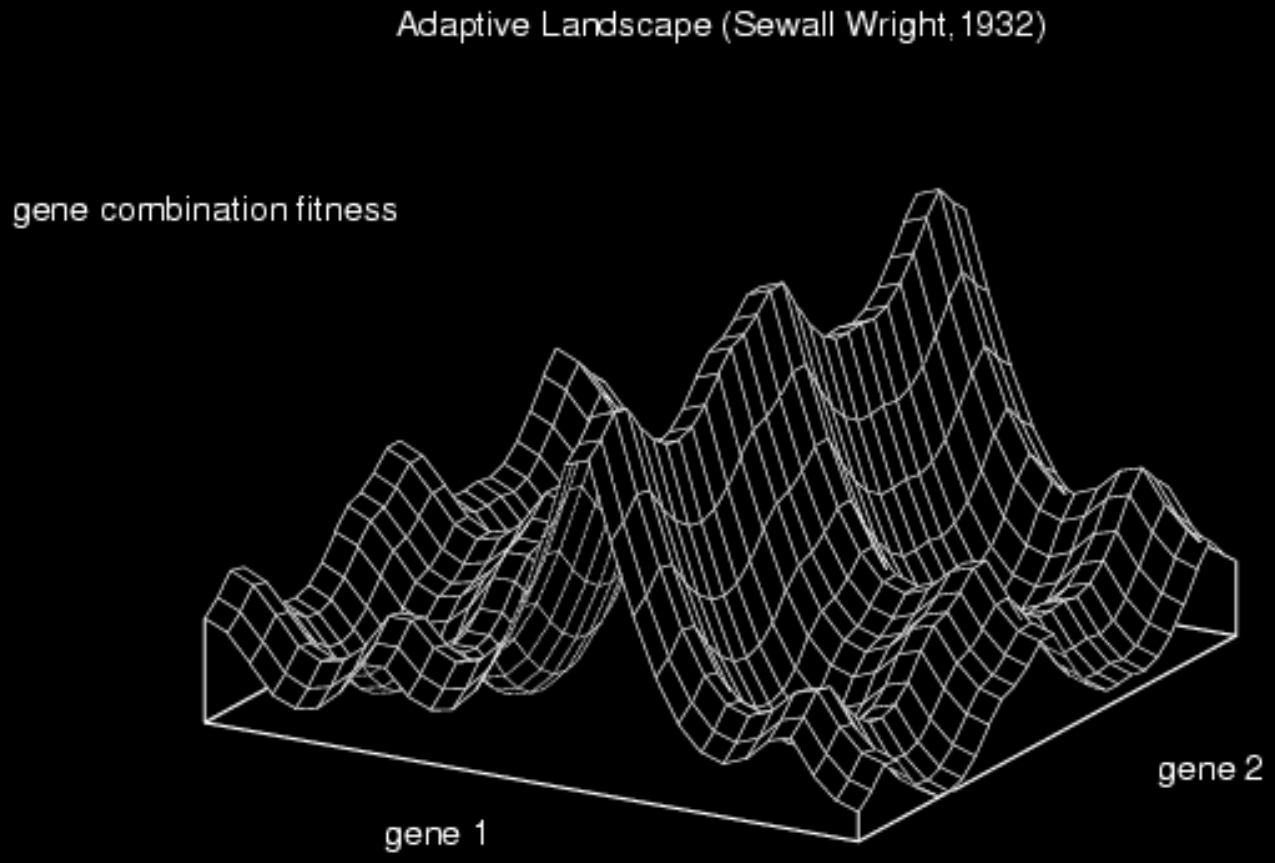
Gen: Secuencia de nucléotidos en la molécula de ADN que contiene información específica para la síntesis de una macromolécula con función celular específica

En el caso de estudio anterior correspondería a distintos objetos a ser observados

Genotipo: Es la totalidad de informacion genética que posee un individuo en forma de ADN. Cuando los individuos se reproducen transfieren parte de su genotipo.

En el caso de estudio correspondería a la secuencia de objetos a ser observados y sus tiempos de observación

Paisaje adaptativo

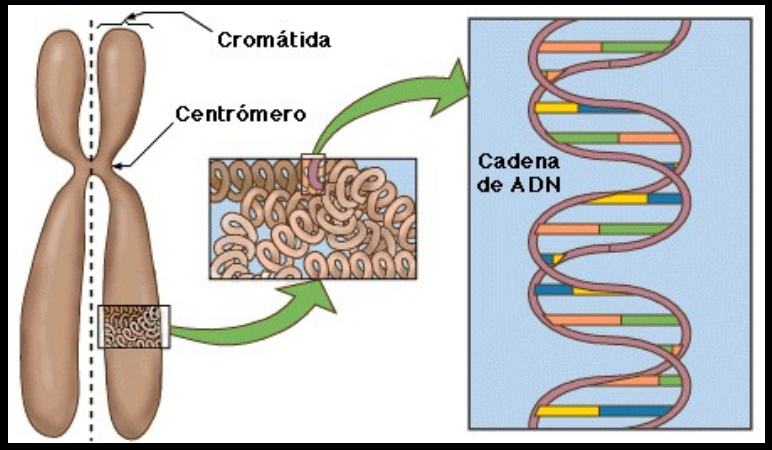


Paisaje adaptativo: Superficie multidimensional que mide la aptitud de los individuos para distintas combinaciones de características genotípicas o fenotípicas.

En la naturaleza, las especies se encontrarían cerca de mínimos locales del paisaje adaptativo. La selección natural puede ocurrir tanto hacia máximos como mínimos por su carácter aleatorio.

En el paralelo computacional, el paisaje adaptativo corresponde a la función objetivo que queremos maximizar.

Reproducción sexual en la naturaleza



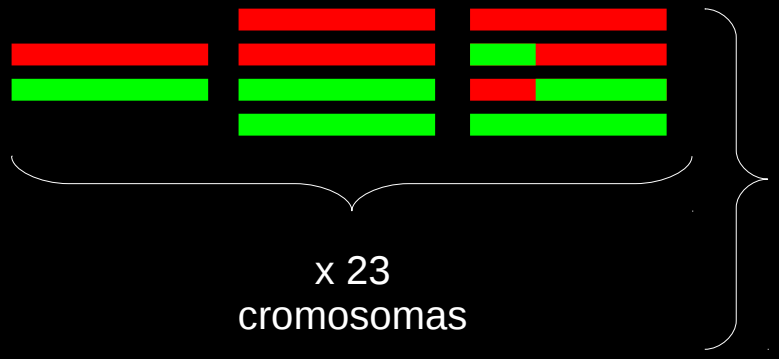
Cromosoma: Antes de la división celular el ADN se organiza en cromosomas. El ser humano forma 23 pares. Cada par está compuesto por una copia idéntica del cromosoma del óvulo de la madre y otra del espermio del padre

Haploide: Célula que contiene una sola componente genética (23 cromosomas en lugar de 23 pares). Espermios y óvulos.

Diploide: Célula que contiene información genética doble, formada al unir información genética de dos haploides: el del padre y el de la madre. La mayoría de las células.

Meiosis:

1. un par de cromosomas (paterno y materno) se alínea físicamente
2. cada cromosoma se duplica para formar 4 cromosomas alineados
3. los dos cromosomas centrales se rompen en un punto aleatorio y se mezclan. El resultado es cuatro cromosomas por par de cromosomas original: un cromosomas idéntico al de la madre, otro al del padre y dos combinaciones nuevas.
4. se forman 4 nuevos haploides usando combinaciones aleatorias de los 23 pares de cromosomas. Los haploides podrán formar diploides en una nueva generación.



4 haploides, cada uno con 23 cromosomas, cada cromosoma elegido al azar entre cuatro cromosomas disponibles

Reproducción sexual en el computador

En el computador el proceso de reproducción sexual es más simple porque es posible generar números aleatorios:

1. se tienen dos soluciones

2. con una probabilidad p_1 se decide *recombinar* o mantener idénticas las soluciones

3. si se recombinan se elige un punto al azar para romper cada solución y crear nuevas soluciones uniendo trozos de soluciones distintas

4. para cada una de las soluciones producidas, se introducen mutaciones aleatorias con una probabilidad p_2

recombinación (*cross-over*)

mutaciones

Reproducción sexual en el computador 2

Un algoritmo genético consiste en la evolución de una población de soluciones de tamaño N, un ejemplo del cual mostramos a continuación:

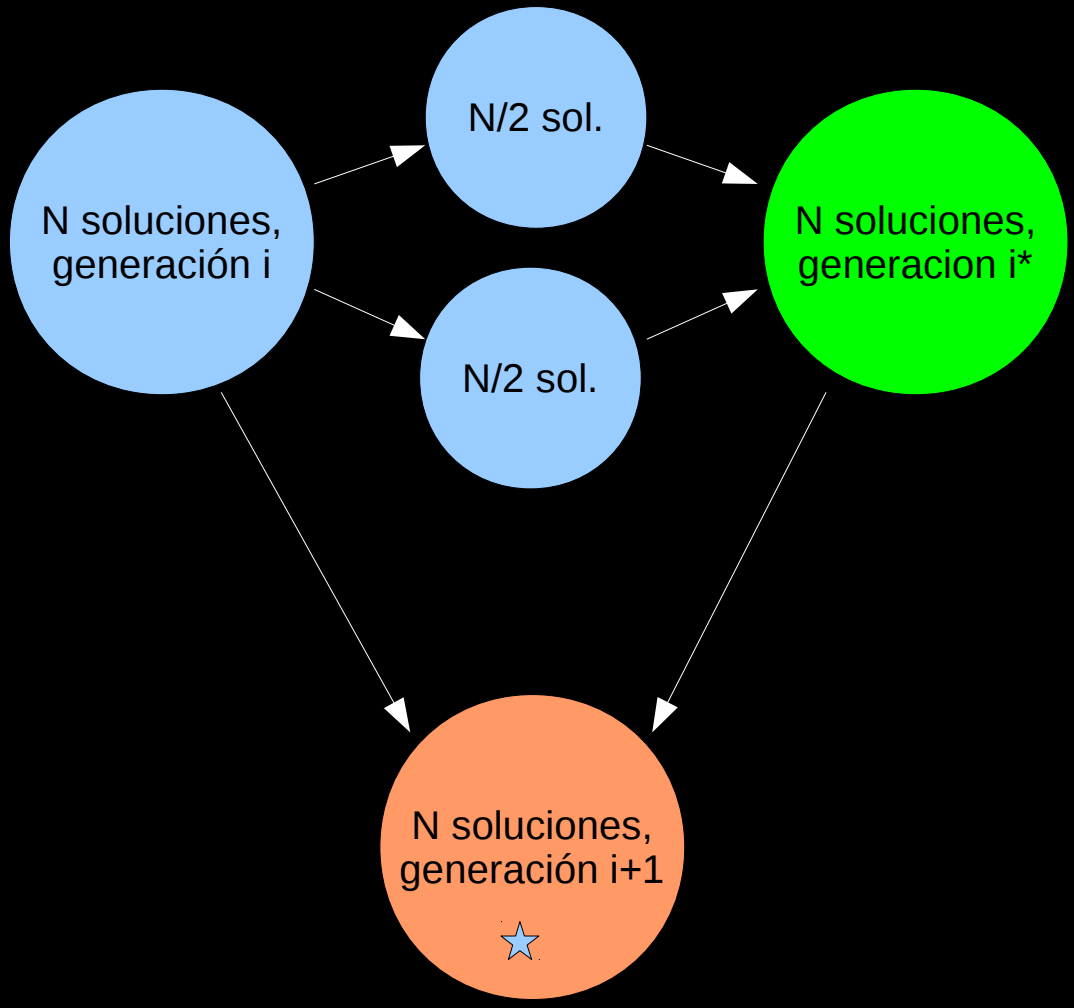
1. Con el objetivo de escoger pares de soluciones para recombinar y mutar se divide la población de tamaño N en dos subpoblaciones de tamaño N/2 de forma aleatoria.

2. El proceso de recombinación y mutaciones se hace formando pares de soluciones, una de cada subpoblación hasta agotar las soluciones disponibles, para producir N nuevas soluciones,

3. Uniendo la población original i y la nueva población i*, ambas de tamaño N, se procede a elegir las mejores N soluciones de entre las 2N soluciones disponibles.

4. se repiten los pasos 1-3 hasta que se cumpla algún criterio de término

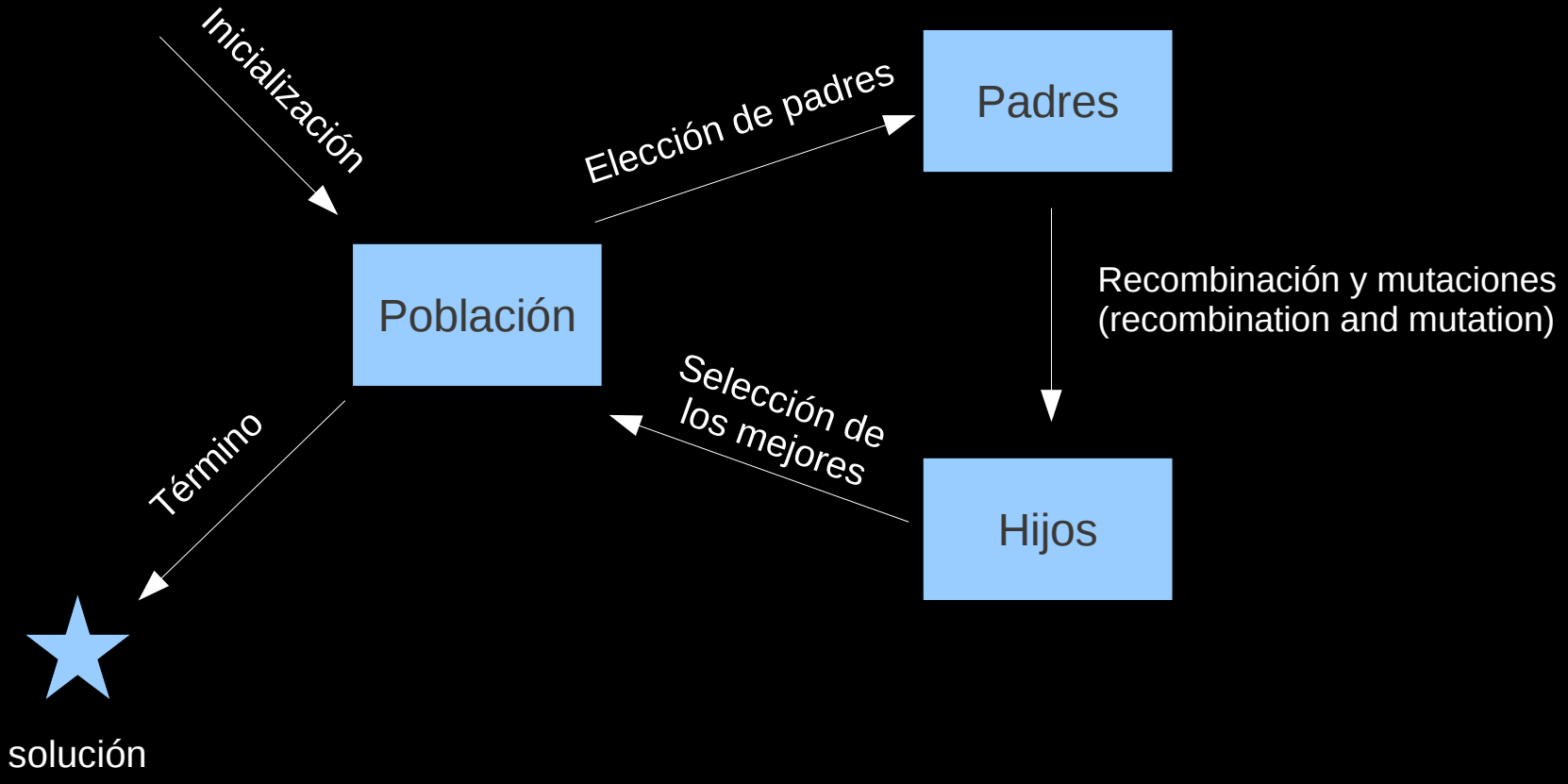
5. Se elige la mejor solución de entre las N soluciones de la última generación



Reproducción sexual en el computador 3

El éxito de un algoritmo genético depende de representar correctamente el problema, de cómo se elige recombinar y mutar las soluciones, de la elección de los parámetros p_1 y p_2, de aplicar correctamente sus restricciones, de cómo se elige la población inicial y de cómo se eligen las mejores soluciones en cada generación, entre otras variables.

La estructura básica de un algoritmo genético es la siguiente:



Implementación

En la práctica los algoritmos genéticos evolucionan una población en las siguientes etapas:

1. las soluciones son distribuidas por todo el espacio de búsqueda de forma aleatoria

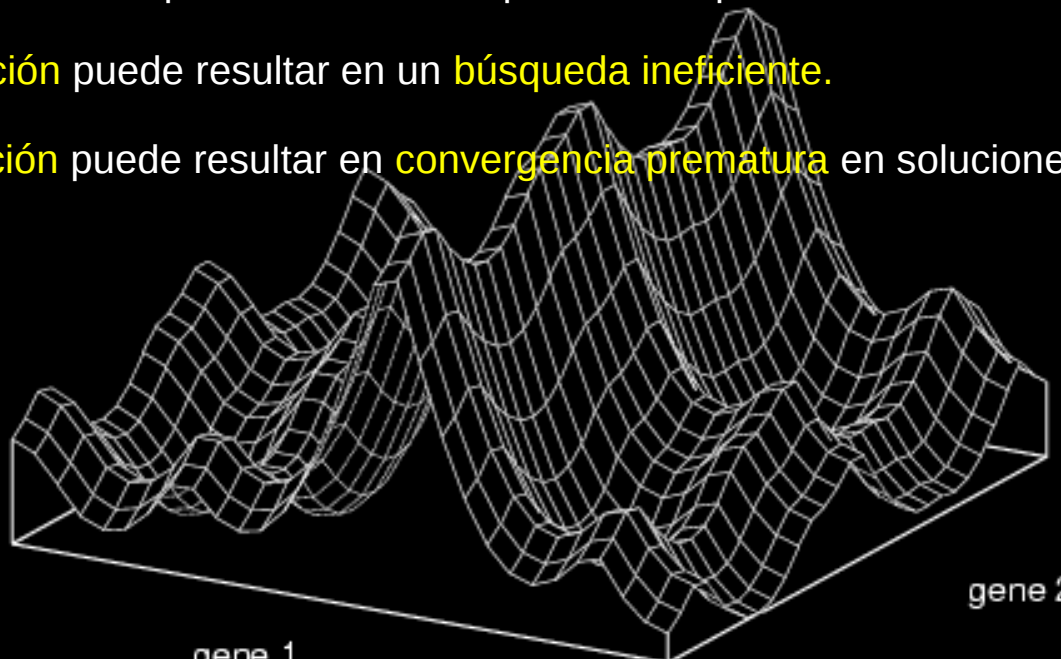
2. luego de algunas generaciones, la población abandona las regiones de baja aptitud y comienza a subir algunos de los máximos locales del problema

3. Si la condición de término está bien definida, la población se concentra en un número pequeño de mínimos locales, algunos de ellos subóptimos. Es posible que la población nunca encuentre un máximo global.

Un buen algoritmo genético logra un equilibrio entre la exploración de nuevas regiones del paisaje adaptativo y la explotación de los máximos locales de éste. Es decir, debe ser capaz de abandonar máximos locales, pero al mismo tiempo no debe perderlos de vista demasiado.

Demasiada exploración puede resultar en un búsqueda ineficiente.

Demasiada explotación puede resultar en convergencia prematura en soluciones subóptimas.



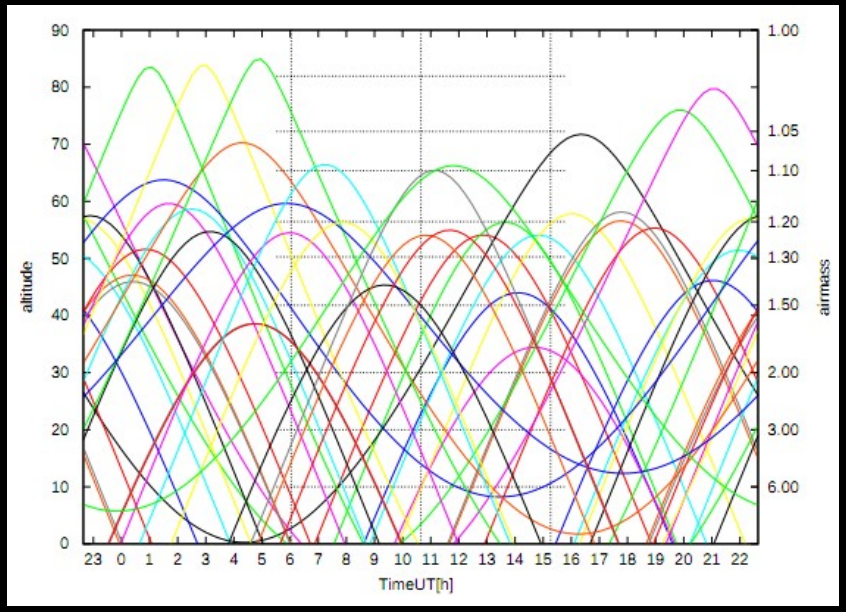
Implementación

Para el caso de estudio (búsqueda de supernovas) cada solución de una población será un plan de observaciones particular.

Como los algoritmos genéticos se basan en una combinación de caracteres discretos, debemos representar nuestro problema de forma discreta.

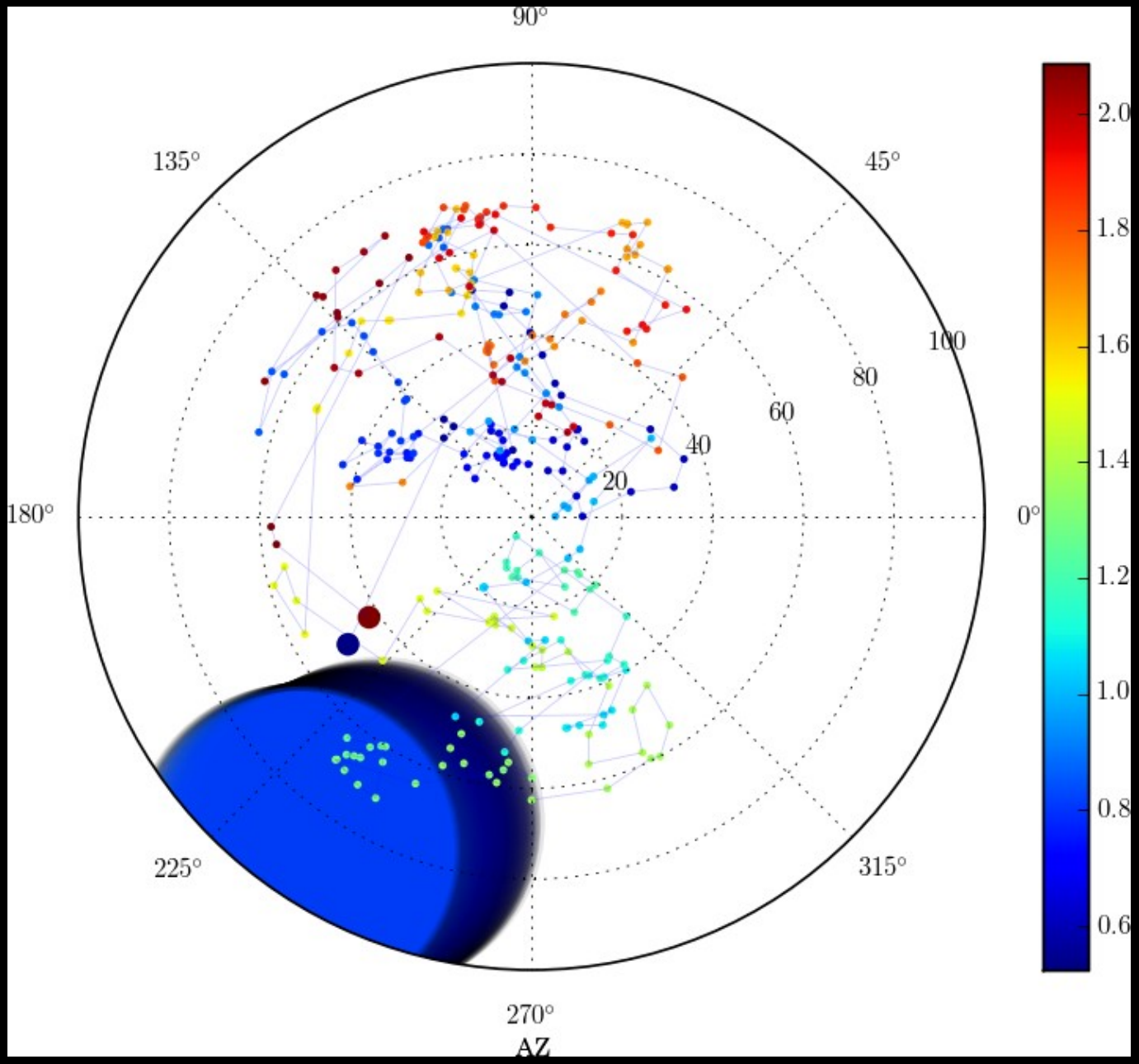
Ejemplo:

Representación 1: Cada plan de observación estará compuesto por una lista de números, cada número representará una galaxia a observar. Existirán N!/(N-n)! posibles soluciones, donde N es el número total de galaxias disponibles y n es el número de galaxias posibles de observar durante una noche.



Representación 2: Cada plan de observación será una lista de galaxias *más* su tiempo de observación, elegido de entre una lista discreta de tiempos. Existirán (Nk)!/(Nk-n)! posibles soluciones, donde ahora k es el número de intervalos de tiempo usados para discretizar la noche.

En ambos casos, la restricción que debe satisfacer cada plan es que todas sus galaxias sean visibles al momento de ser observadas.



Ejemplo de plan de observaciones para una noche dada obtenida usando un algoritmo genético. Los colores indican el tiempo, desde azul a rojo. La posicón de la Luna está indicada por el círculo azul grande, que desaparece cuando la luna se pone debajo del horizonte. El plan generado tomó un minuto de cálculo usando un módulo para python escrito en FORTRAN 90.

Representación, recombinación, mutación y selección de individuos

Representación

Es el primer paso y el más importante al momento de diseñar un algoritmo genético

Implica definir el genotipo y su forma de almacenamiento, así como el mapeo entre genotipo y fenotipo (función objetivo) correctamente.

Existen muchas formas de representar cada problema, algunas representaciones comunes son:

1. Representación binaria: la solución se representa como una secuencia de 1s y 0s *explícitamente.* Si bien es conceptuamente sencilla, puede ser difícil de interpretar en la mayoría de los problemas.

2. Representación entera o de punto flotante: la solución se representa como una lista de enteros o de números de punto flotante. El número de bits asociado a cada entero definirá el número de soluciones posibles. Si el valor de las variables no está restringido y el número de bits de representación es muy grande el algoritmo puede sufrir de un exceso de posibles soluciones.

3. Representación por permutaciones: Cuando se trata de elegir el orden en que una serie de eventos debe ocurrir la representación más natural es una secuencia de índices en orden de ocurrencia. Así como el orden de ocurrencia puede ser importante (*job shop scheduling*), la distancia entre las soluciones también puede ser importante (*travelling salesman problem*)

Recombinación

Existen muchas forma de realizar la recombinación, que dependen de la representación del problema. Entre ellas se encuentran:

1. Representación binaria: a) *1-point cross-over*, las soluciones se quiebran en un mismo bit para intercambiar partes, b) *N-point cross-over*, se usan varias posiciones para quebrar y agregar soluciones, c) *uniform cross-over,* las solucione se quiebran en todos sus bits para formar una nueva solución

2. Representación entera: Análogo a la representación binaria

3. Representación de punto flotante: a) operaciones análogas a las de representación binaria, b) las soluciones se quiebran en una posición y en lugar de intercambiar regiones se forman nuevas regiones por operaciones aritméticas, e.g. promediar los valores a la derecha de la posición de quiebre

4. Representación por permutaciones: a) operaciones análogas a las de representación binaria, b) permutaciones que intentan preservar las propiedades de adyacencia de las soluciones de los padres, muchas soluciones posibles

5. Recombinación multi-padre: Se recombinan más de dos padres para crear hijos.

Mutación

Existen muchas formas de realizar una mutación, que también dependen de la representación de la solución. Entre las más comunes se encuentran:

1. Representación binaria: a) cambiar el valor de un bit elegido aleatoriamente, b) intercambiar el valor de un par de bits elegidos aleatoriamente.

2. Representación entera: a) cambiar el valor de un entero de forma aleatoria, b) se perturba un entero por una cantidad pequeña

3. Representación de punto flotante: a) se cambia el valor de un número elegido aleatoriamente en un intervalo de valores, b) se perturba un número por una cantidad pequeña

4. Representación por permutaciones: a) se intercambian los índices de dos elementos de la solución, b) se cambia la posición de un elemento aleatoreamente, c) se reordena aleatoreamente una región de la solución, d) se invierte el orden de una región de la solución

Selección de individuos

Además de cómo representar, combinar y mutar las soluciones, se debe elegir una forma para seleccionar los padres a reproducir (no siempre se reproduce toda la población), así como los sobrevivientes para la próxima generación.

Elegir soluciones exclusivamente en términos de su función objetivo puede disminuir la diversidad de las poblaciones y llevar a convergencia prematura. Para evitar eso existen las siguiente alternativas:

1. Elección probabilística: como una función del ranking de las soluciones de la población.

2. Elección por competencia: las soluciones compiten entre sí en un campeonato. Esto evita tener que elaborar un ranking cuando esto es caro computacionalmente

3. Elitismo: Se mantiene una fracción de las mejores soluciones de la generación anterior artificialmente (solución adoptada en el caso de estudio). Se utiliza para aumentar la eficiencia de la búsqueda, pero puede llevar a convergencia prematura.

4. Eliminación soluciones malas o muy viejas: En algunos casos se recomienda eliminar malas soluciones o aquellas que han sobrevivido por muchas generaciones. Un exceso de eliminación de soluciones malas puede llevar a convergencia prematura y un exceso de eliminación de soluciones viejas puede llevar a búsqueda ineficiente.

Conclusiones

El problema de planificar observaciones es un problema complejo.

Es posible cuantificar de forma objetiva la calidad de un plan de observaciones usando la ecuación de señal a ruido y probabilidades.

Los métodos de solución de problemas complejos más usados están inspirados en sistemas biológicos.

Es posible resolver el problema de *scheduling* utilizando el método de optimización basado en algoritmos genéticos.

Es importante representar correctamente el problema para encontrar una buena solución.

Además de la representación, es importante elegir una forma de recombinación, mutación y selección apropiadas.

Es posible introducir sesgos en el proceso de reproducción (siguiente clase)

Referencias

“Introduction to Evolutionary Computing”, Agoston E. Eiben, J.E. Smith. Springer

“Ant Colony Optimization”, Marco Dorigo, Thomas Statzle. Springer

“Scheduling in Targeted Transient Surveys and a New Telescope for CHASE”, F. Förster, N. López, J. Maza, P. Kubánek, G. Pignata

F2PY: Fortran to Python interface generator, http://cens.ioc.ee/projects/f2py2e/