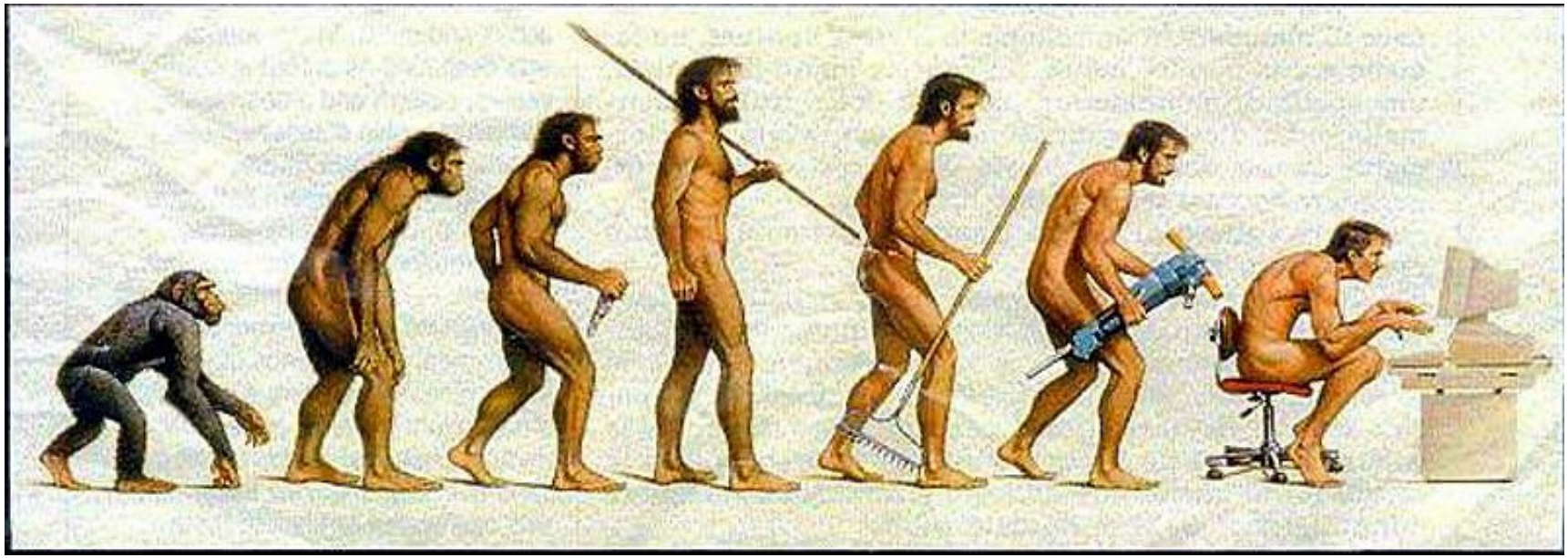


Aplicando AGs a problemas

Héctor Alejandro Montes
hamontesv@uaemex.mx
http://scfi.uaemex.mx/hamontes

Advertencia

No use estas diapositivas como referencia única de estudio durante este curso. La información contenida aquí es sólo una guía para las sesiones de clase y de estudio futuro. Para obtener información más completa, refiérase a la bibliografía dada durante la presentación del curso.

Genetic Algorithms

"Genetic Algorithms are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you might not otherwise find in a lifetime."

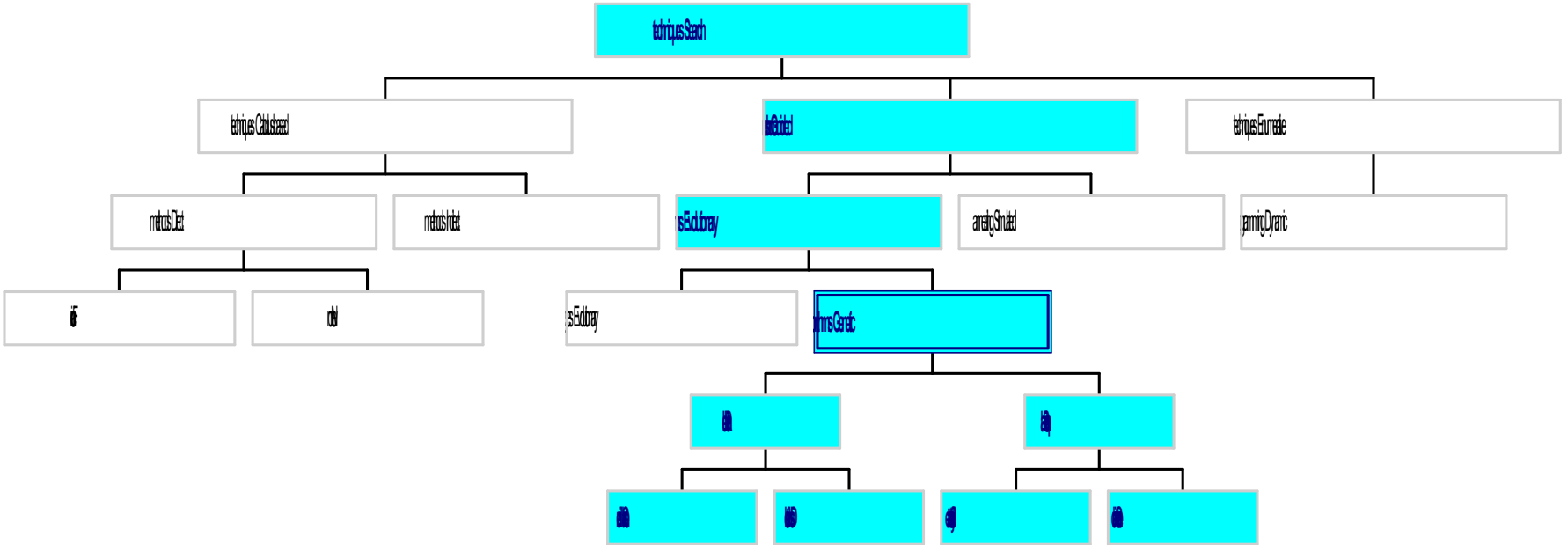
- Salvatore Mangano

Computer Design, May 1995

Aplicando AG a un problema

- Resolver problemas básicos de implementación:
  - Representación
  - Tamaño de población, probabilidad de cruza y mutación
  - Método de selección
  - Operadores de cruza y mutación
- Criterio de paro
- Desempeño y escalabilidad
- Función fitness (a menudo la parte más difícil)

Técnicas de búsqueda



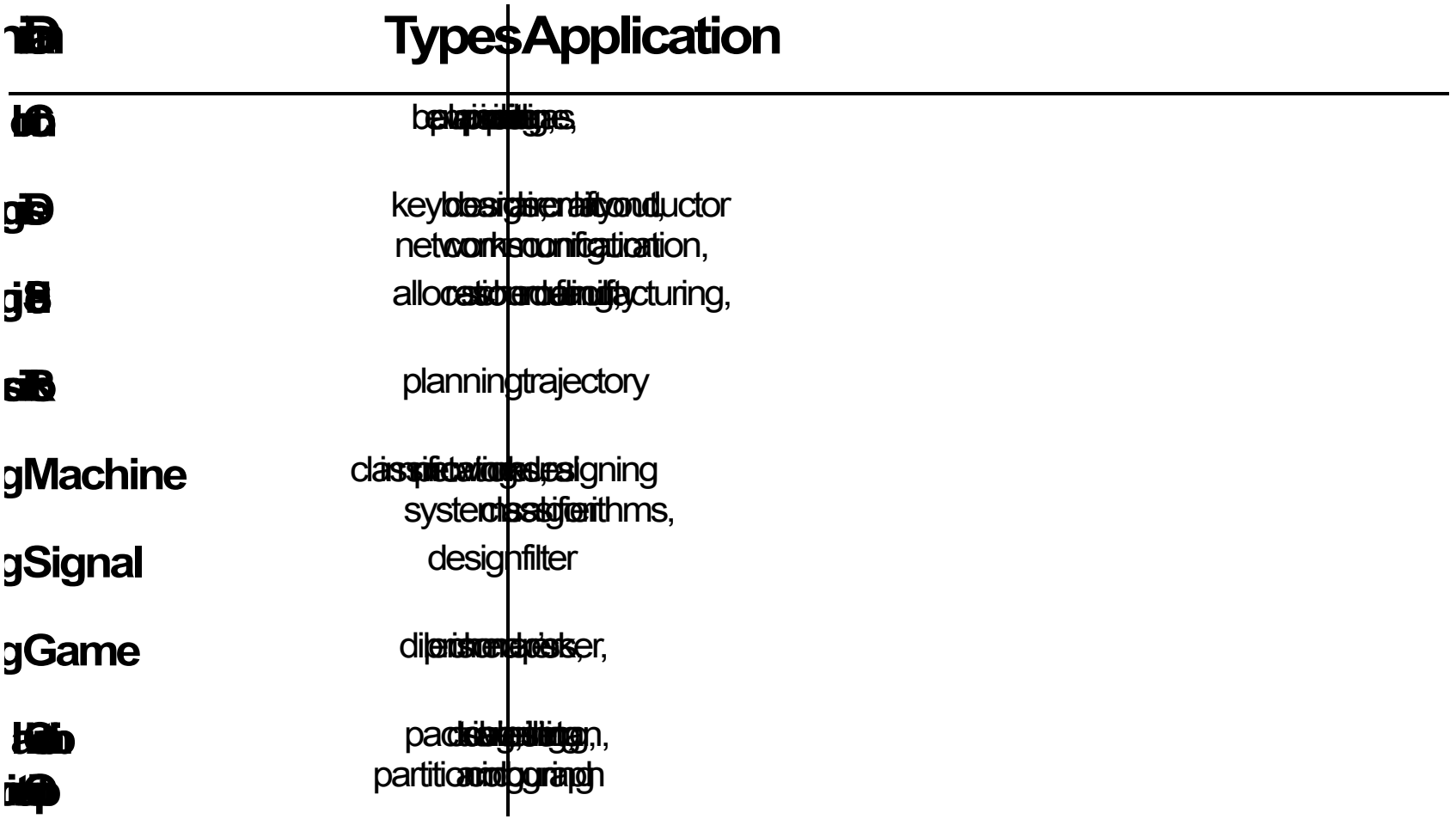
Beneficios de las AGs

- El concepto es fácil de entender
- General e independiente del problema
- Adecuados para ambientes "*ruidosos*"
- Siempre obtenemos una respuesta que se espera mejore con el tiempo
- Fácilmente paralelizables y distribuíbles
- Útiles en optimización multi-objetivo

Cuándo usarlos

- Cuando soluciones alternativas son muy lentas o muy complicadas
- Cuando el problema es similar a otro en el que se haya tenido éxito
- Cuando se observen beneficios evidentes en la búsqueda basada en poblaciones

Algunas aplicaciones



GA & Microsoft

"Almost eight years ago ... people at Microsoft wrote a program [that] uses some genetic things for finding short code sequences. Windows 2.0 and 3.2, NT, and almost all Microsoft applications products have shipped with pieces of code created by that system."

- Nathan Myhrvold, Microsoft Advanced Technology Group, *Wired*, September 1995

Problemas

- *El problema del agente viajero*
- Planificación de trayectorias para robots
- El problema de programación de procesos (*job-shop scheduling*)
- El problema de satisfactibilidad (*3-SAT*)
- *The bin packing problem*
- El problema de horarios (*timetable*)
- Problema "*El dilema del prisionero*"

Planificación de trayectorias para robots

- Dada una posición y orientación inicial ***p*** de un robot, crear una trayectoria que lo lleve a una posición y orientación final ***q***
- La trayectoria (serie de movimientos) para llevar al robot de ***p*** a ***q***, describe también las velocidades y acelaraciones del robot como una función de tiempo

Job-shop scheduling

- *Job-shop* es una instalación de manufactura organizada en procesos.
- Un Job-shop genera productos utilizando uno o más planes alternativos de procesos que requieren recursos y tardan cierto tiempo en cada paso.
- El problema es seleccionar una secuencia de operaciones junto con una asignación de tiempos de inicio/fin y recursos para cada operación.

Satisfactibilidad (3-SAT)

- Es el problema de determinar si las variables de una formula booleana pueden ser asignadas de tal manera que sea evaluada como TRUE.
- El problema es complejo aun cuando todas las expresiones tengan *forma normal conjuntiva* con 3 variables por cláusula (el problema 3-SAT):

  (x11 OR x12 OR x13) AND (x21 OR x22 OR x23) AND (x31 OR x32 OR x33) AND ...

- Donde cada variable ***x*** es una variable o una negación de una variable y cada variable puede aparecer multiples veces en la expresión.

Bin packing

- Objetos de diferentes volúmenes deben ser empacados en un número finito de contenedores de capacidad ***C*** de tal forma que se utilicen el número mínimo de contenedores.

El problema de horarios

- Existe una lista ***P*** de profesores, una lista de horarios ***H*** y una lista de clases ***C***. El problema es encontrar una combinación de horarios ***P-H-C*** que cumpla los criterios de optimalidad requeridos.
- Los criterios pueden ser didácticos (distribuir algunas clases durante la semana), objetivos personales (profesores de asignatura), u objetivos institucionales (un profesor no puede tenr dos cursos iguales).

El dilema del prisionero

- Dos sospechosos son arrestados y la policía tiene insuficiente evidencia para condenarlos por lo que deciden interrogarlos por separado. Si uno de los prisioneros testifica contra el otro y el otro permanece callado, el traidor es liberado y el otro recibe 10 años de prisión. Si ambos permanecen callados, los dos son sentenciados a sólo 6 meses de cárcel. Si cada prisionero traiciona al otro, cada uno recibe 5 años de cárcel.
- Cada prisionero debe elegir entre testificar contra el otro o permanecer callado. A cada prisionero se le garantiza que el otro no sabrá de una posible traición antes de que la investigación termine. ¿Cómo deberían actuar los prisioneros?.
- *El dilema del prisionero* es un problema fundamental en teoría de juegos que demuestra porqué dos personas pueden no cooperar entre sí aún y cuando esta en su interés hacerlo.

Procedimiento

- Presentaciones sobre el problema.
  - Entregable: *reporte*
- Presentaciones de cómo lo resolverá
  - Entregable: *reporte*
- Proyecto final
  - *Paper* presentando resultados

Consideraciones importantes

- Nunca emitir conclusiones de una única ejecución
  - Utilizar medidas estadísticas (medias, medianas, …)
  - Con un número suficiente de ejecuciones independientes
- “He obtenido la solución en una sóla experimentación sobre un problema”
  - No se debe asumir la actuación de un algoritmo sobre ejemplos simples si se desea trabajar con casos reales
- Desde el punto de vista de las aplicaciones:
  - Encontrar *una* solución de *buena calidad* al menos una vez
  - Encontrar *al menos una* solución de *buena calidad* en cada ejecución

Problemas

- *El problema del agente viajero*
- Planificación de trayectorias para robots
- El problema de programación de procesos (*job-shop scheduling*)
- El problema de satisfactibilidad (*3-SAT*)
- *The bin packing problem*
- El problema de horarios (*timetable*)
- Problema "*El dilema del prisionero*"