هوش مصنوعی

حسین کارشناس

دانشکده ریاضی

ترم اول ۹۴ – ۹۳

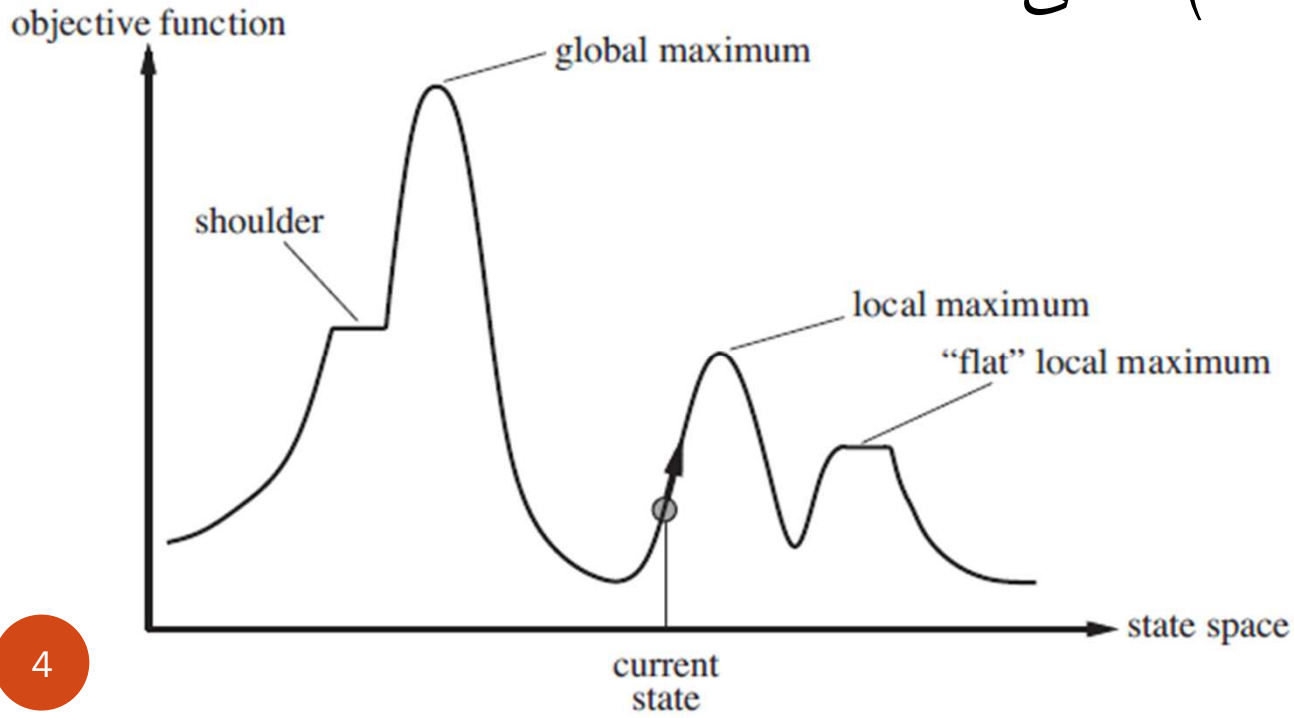
فراتر از جستجوی کلاسیک

فراتر از جستجوی کلاسیک

- فرضیات در جستجوی کلاسیک
  - محیط‌های قطعی، مشاهده‌پذیر و شناخته شده ← راه‌حل یک دنباله ثابت از کنش‌هاست
- جستجوی موضعی (local) در مقابل جستجوی نظام‌مند (systematic)
  - عدم نیاز به نگهداری مسیرهای مختلف و حالت‌های مشاهده شده
    - فقط گره‌ی فعلی (current node) در نظر گرفته می‌شود
    - در بسیاری از مسائل مسیر رسیدن به هدف مهم نیست
      - مسأله ۸ ملکه شطرنج
  - مزایا
    - نیاز به حافظه بسیار کم (معمولاً ثابت)
    - یافتن راه‌حل‌های معقول (reasonable) برای مسائلی با فضاهای حالت بزرگ یا نامتناهی

جستجوی موضعی

- تفاوت‌ها نسبت به چهارچوب تعریف قبلی برای مسائل جستجو
  - استفاده از تابع هدف (objective function) بجای آزمایش هدف
    - مسائل بهینه‌سازی (optimization)
- چشم‌انداز (landscape) فضای حالت
  - بهینه سراسری
  - کمال و بهینگی



جستجوی تپه‌نوردی (hill-climbing)

- بررسی تمام حالات مجاور (neighbor) حالت فعلی و انتخاب بهترین

```
function HILL-CLIMBING(problem) returns a state that is a local maximum

  current ← MAKE-NODE(problem.INITIAL-STATE)
  loop do
    neighbor ← a highest-valued successor of current
    if neighbor.VALUE ≤ current.VALUE then return current.STATE
    current ← neighbor
```

- با رسیدن به یک قله (دره) متوقف می‌شود
  - درخت جستجو را نگهداری نمی‌کند
  - حالات فراتر از حالات مجاور را در نظر نمی‌گیرد: جستجوی موضعی حریصانه
- تبدیل دیدگاه‌های تندترین صعود (steepest ascent) و شیب نزولی (gradient descent) به یکدیگر در بهینه‌سازی

جستجوی تپه‌نوردی

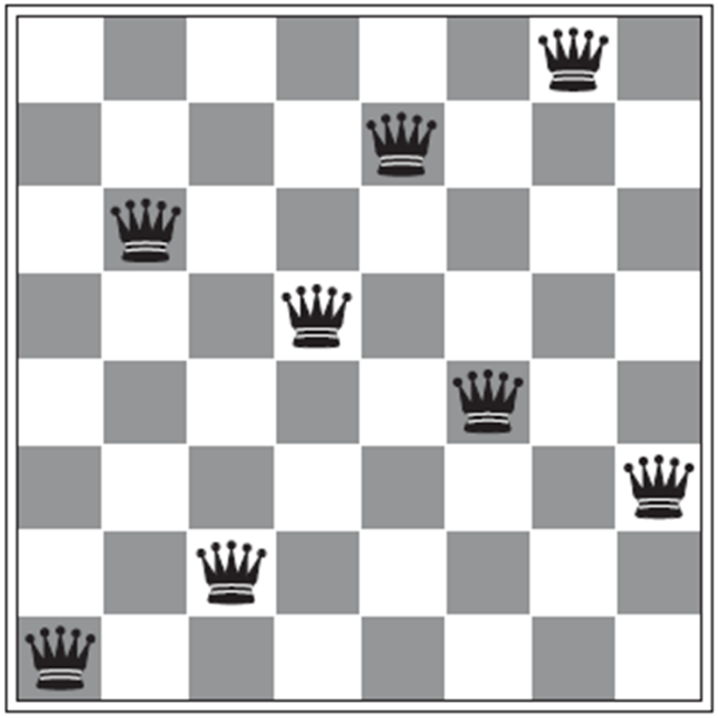
- مثال: ۸ ملکه شطرنج

h = 17

18	12	14	13	13	12	14	14
14	16	13	15	12	14	12	16
14	12	18	13	15	12	14	14
15	14	14	♛	13	16	13	16
♛	14	17	15	♛	14	16	16
17	♛	16	18	15	♛	15	♛
18	14	♛	15	15	14	♛	16
14	14	13	17	12	14	12	18

در ۵ قدم

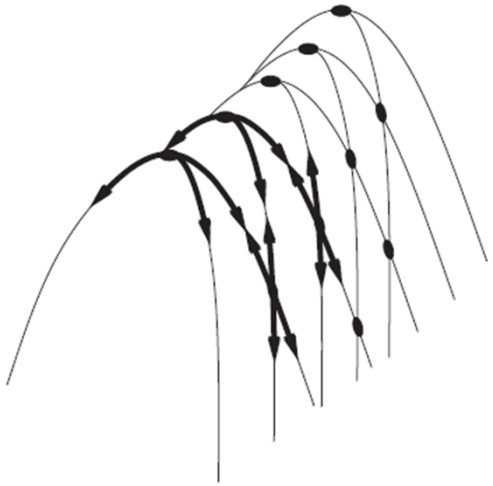
h = 1



- پیشرفت سریع به سمت هدف و سهولت بهبود حالات بد

جستجوی تپه‌نوردی

- الگوریتم ممکن است با مشکل مواجه شود
  - بهینه‌های موضعی (local optima)
  - لبه‌ها – شیارها (ridges) ←



  - سطوح مسطح (plateaux)
    - بهینه‌های موضعی یا شانه‌ها
    - استفاده از حرکت‌های به کنار (sideways move): در نظر گرفتن یک بیشینه
- گونه‌های مختلف الگوریتم
  - تپه‌نوردی تصادفی (stochastic): احتمال حالت بعدی بسته به تندی شیب
  - تپه‌نوردی اولین گزینه (first-choice)

جستجوی تپه‌نوردی

- کامل نیستند ← معمولاً در بهینه‌ی موضعی گیر می‌کنند
- تپه‌نوردی با شروع مجدد تصادفی (random-restart)
  - یک دنباله از تپه‌نوردی‌ها با شروع از حالت‌های اولیه‌ی تصادفی تا یافتن هدف
  - احتمال کامل بودن آن به یک میل می‌کند (با افزایش تکرارها)
    - تعداد شروع‌های مجدد مورد انتظار: $1/p$
      - p احتمال موفقیت هر تپه‌نوردی
    - تعداد قدم‌های مورد نیاز: $N_s + \frac{1-p}{p} N_f$
      - N_s متوسط قدم‌ها در تکرار موفقیت‌آمیز
      - N_f متوسط قدم‌ها در تکرارهای ناموفق
  - کارایی بسته به وضعیت چشم‌انداز مسأله دارد (تعداد بهینه‌های موضعی)

تبرید شبیه‌سازی شده (Simulated Annealing)

- الگوریتم تپه‌نوردی حرکتی بر خلاف تابع هدف نمی‌کند
  - کارایی بالایی دارد اما کامل نیست
- گام‌برداری تصادفی (Random Walk): حالت بعدی کاملاً بصورت تصادفی انتخاب می‌شود
  - کارایی بسیار ضعیفی دارد اما کامل است
- تلفیق این دو روش برای داشتن دو مزیت کارایی و کمال
  - فرار از بهینه‌های موضعی با قبول حرکت‌های بد که به مرور بسامد آنها کاهش می‌یابد
- مفهوم تبرید: گرم کردن و سپس سرد کردن تدریجی
  - کمینه کردن انرژی جنبشی

تبرید شبیه‌سازی شده

- مانند حرکت یک توپ پینگ پونگ بر روی یک سطح نامسطح برای رسیدن به عمیق‌ترین شکاف با کمک لرزاندن
- کاهش تدریجی میزان لرزاندن (دما) ← نیاز به یک زمانبندی (schedule) دما

function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
  inputs: *problem*, a problem
    schedule, a mapping from time to "temperature"

  current ← MAKE-NODE(*problem*.INITIAL-STATE)
  for $t = 1$ **to** ∞ **do**
    $T \leftarrow schedule(t)$
    if $T = 0$ **then return** *current*
    next ← a randomly selected successor of *current*
    $\Delta E \leftarrow next.\text{VALUE} - current.\text{VALUE}$
    if $\Delta E > 0$ **then** $current \leftarrow next$
    else $current \leftarrow next$ only with probability $e^{\Delta E/T}$

جستجوی پرتوی موضعی (local beam)

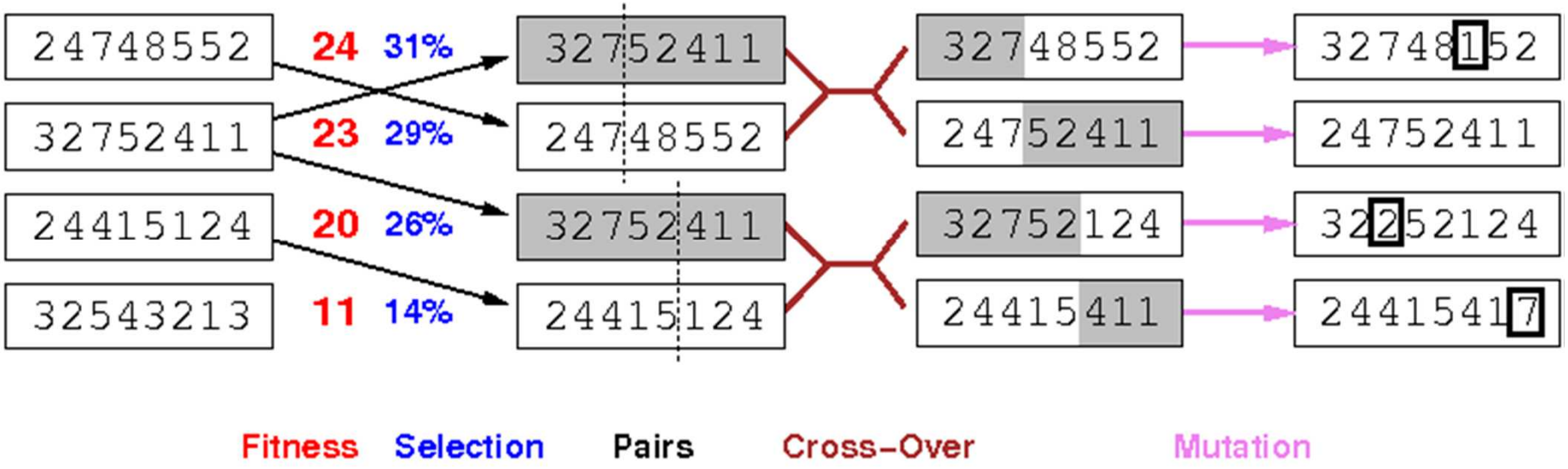
- نگهداری و بررسی همزمان k حالت
  - انتخاب بهترین k حالت از بین تمام همسایگان تمام k حالت قبلی
- تفاوت با k جستجوی تپه‌نوردی با شروع مجدد تصادفی که به صورت همزمان اجرا شوند
  - امکان تبادل اطلاعات بین جستجوهای موازی در جستجوی پرتوی موضعی
- امکان همگرایی زودرس
  - جستجوی پرتوی تصادفی (stochastic)
    - انتخاب k حالت از بین تمام همسایگان با احتمالی که به کیفیت آنها بستگی دارد
  - شباهت به انتخاب طبیعی (natural selection)

الگوریتم‌های ژنتیکی (Genetic Algorithms)

- جمعیتی از حالت‌ها یا افراد (individuals) را نگهداری می‌کند
  - مشابه الگوریتم جستجوی پرتوی موضعی
  - نمایش هر فرد با رشته‌ای از حروف متعلق به یک الفبای محدود
    - مثال: رشته‌های بیتی
- ارزیابی حالت‌ها (افراد) با یک تابع برازندگی (fitness function)
- عملگرهای ژنتیکی: استفاده از دو عملگر اصلی برای تولید افراد جدید
  - تقطیع (crossover)
  - جهش (mutation)
- انتخاب طبیعی
  - انتخاب افراد بر اساس برازندگی برای بازتولید (reproduction) افراد جدید

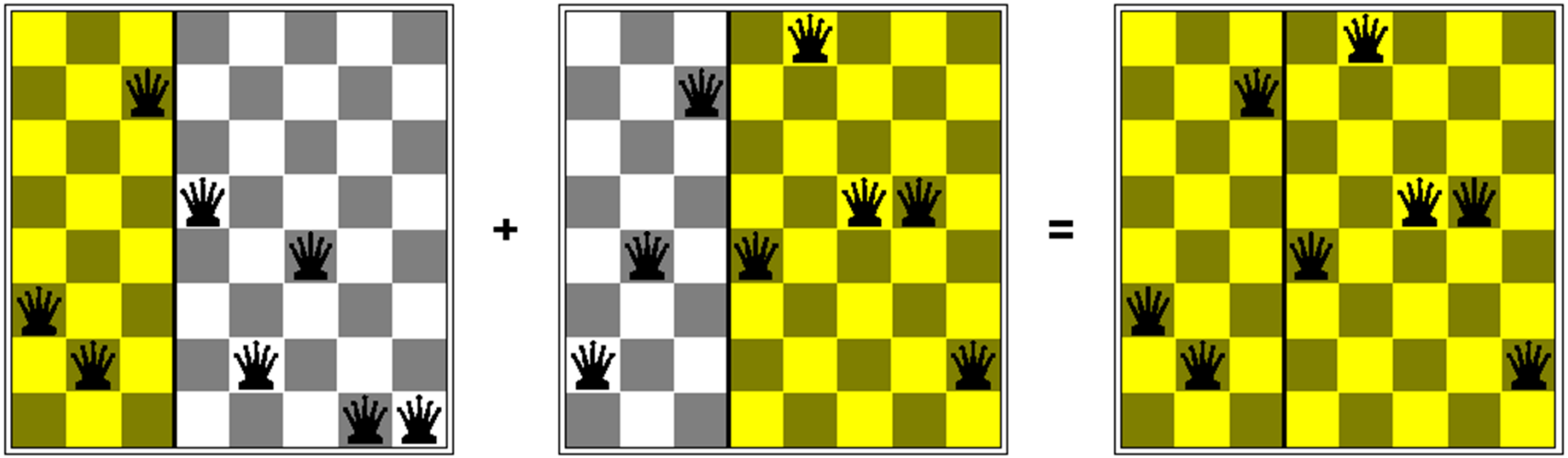
الگوریتم‌های ژنتیکی

- مثال: ۸ وزیر



- پارامترهای تصادفی: احتمال تقطیع، نقطه تقطیع، احتمال جهش، نقطه جهش

الگوریتم‌های ژنتیکی



- تقطیع می‌تواند سطح پیش‌روی (granularity) جستجو را افزایش دهد
  - فاصله والدین به مرور نسبت به هم کم می‌شود
- تئوری شِما (schema theory) برای تحلیل عملکرد الگوریتم
- در بهینه‌سازی بسیار پرکاربرد است