**Các giải thuật tìm kiếm mù**

**- Tìm kiếm rộng (Breath-First-Search)**

BFS là 1 phương pháp duyệt đồ thị hoặc cây bằng cách truy cập tất cả các đỉnh cùng cấp trước khi đi sâu vào các đỉnh cấp sau. Cách hoạt động:

1. Bắt đầu từ 1 đỉnh bất kỳ trong đồ thị hoặc cây.
2. Thêm đỉnh bắt đầu vào hàng đợi.
3. Lặp cho đến khi hàng đợi trống:
   * Lấy đỉnh ở đầu hành đợi ra.
   * Đánh dấu đỉnh đã được duyệt qua.
   * Lặp qua tất cả các đỉnh kể với đỉnh hiện tại: Nếu đỉnh kề chưa được duyệt, thêm nó vào hàng đợi.
4. Lặp lại quá trình cho tất cả các đỉnh chưa được duyệt.

BFS đảm bảo rằng các đỉnh được thăm theo thứ tự từ xa đến gần, tức là duyệt qua các đỉnh cách đỉnh bắt đầu 1 cấp, sau đó duyệt qua các đỉnh cách đỉnh bắt đầu 2 cấp và tiếp tục như vậy. BFS thường được sử dụng trong việc: tìm kiếm đường đi gần nhất, kiểm tra liên thông của đồ thị, tìm kiếm theo cấp bậc.

**- Tìm kiếm sâu (Depth-First-Search)**

DFS là 1 thuật toán dùng để duyệt đồ thị hoặc cây. Ở mỗi bước, nó lần lượt đi sâu vào 1 nhánh của đồ thị cho đến khi không thể đi tiếp nữa, sau đó quay lại và thử nhánh khác. Cách hoạt động:

1. Bắt đầu từ đỉnh bất kỳ trong đồ thị hoặc cây.
2. Đánh dấu đỉnh hiện tại là đã duyệt.
3. Lặp lại qua tất cả các đỉnh kề với đỉnh hiện tại: Nếu đỉnh kề chưa được duyệt, thực hiện DFS từ đỉnh đó.
4. Lặp lại quá trình cho tất cả các đỉnh chưa được duyệt.

DFS thường được thực hiện bằnh cách đệ quy, nhưng cũng có thể sử dụng ngăn xếp để lưu trữ và duyệt các đỉnh. Thường được sử dụng trong nhiều ứng dụng như: tìm kiếm đường đi, kiểm tra liên thông của đồ thị, tìm kiếm các thành phần liên thông mạnh.

**Các kỹ thuật tìm kiếm có thông tin (tìm kiếm Heuristic)**

**- Leo đồi (Hill climbing)**

Hill climbing là 1 phương pháp tìm kiếm cục bộ dựa trên việc di chuyển từ 1 trạng thái sang trạng thái khác để tìm kiếm 1 giải pháp tốt nhất. Cách hoạt động:

1. Bước khởi tạo: Bắt đầu từ 1 trạng thái bất kỳ.
2. Đánh giá: Đánh giá mức độ tốt của trạng thái hiện tại bằng 1 hàm đánh giá.
3. Tìm kiếm lân cận: Liệt kê tất cả các trạng thái lân cận có thể đạt được từ trạng thái hiện tại bằng các phép biến đổi nhỏ.
4. Chọn lựa: Chọn trạng thái lân cận có hàm đánh giá tốt nhất.
5. Lặp lại: Lặp lại quá trình từ bước 2 với trạng thái mới được chọn.
6. Kiểm tra dừng: Dừng khi không có trạng thái lân cận nào có hàm đánh giá tốt hơn trạng thái hiện tại hoặc khi đạt được 1 điều kiện dừng khác.

**- Háu ăn (Greedy)**

Greedy là 1 phương pháp giải quyết vấn đề bằng cách luôn chọn lựa giải pháp tốt nhất ở mỗi bước hiện tại mà không xem xét những hậu quả xa hơn trong tương lai. Cách hoạt động:

1. Bước khởi tạo: Bắt đầu từ 1 điểm bất kỳ.
2. Chọn lựa tốt nhất: Tại mỗi bước, chọn lựa chọn tối ưu nhất dựa trên tiêu chí cụ thể của vấn đề.
3. Thực hiện: Thực hiện lựa chọn đã chọn.
4. Cập nhật trạng thái: Cập nhật trạng thái của bàu toán dựa trên lựa chọn đã thực hiện.
5. Kiểm tra điều kiện dừng: Kiểm tra xem đã đạt được điều kiện dừng hay chưa.

**- Tìm kiếm tốt nhất đầu tiên (Best-First-Search)**

Tìm kiếm tốt nhất đầu tiên là 1 thuật toán tìm kiếm trong không gian trạng thái bằng cách luôn chọn trạng thái có giá trị có giá trị đánh giá tốt nhất trong số các trạng thái lân cận. Cách hoạt động:

1. Bước khởi tạo: Bắt đầu từ trạng thái ban đầu.
2. Đánh giá: Đánh giá mức độ tốt của mỗi trạng thái.
3. Tìm kiếm lân cận: Liệt kê tất cả các trạng thái lân cận có thể đạt được từ trạng thái hiện tại.
4. Chọn lựa tốt nhất: Chọn trạng thái lân cận có giá trị đánh giá tốt nhất.
5. Lặp lại: Lặp lại quá trình từ bước 2 với trạng thái mới được chọn.
6. Kiểm tra dừng: Dừng khi đạt được trạng thái mục tiêu hoặc khi không còn trạng thái mới nào để thử.

- Giải thuật A\*

Giải thuật A\* là 1 thuật toán tìm kiếm đường đi trong đồ thị hoặc trong không gian trạng thái­­­­­­, thường được sử dụng trong bài toán tìm kiếm đường đi ngắn nhất. Cách hoạt động:

1. Bước khởi tạo: Bắt đầu từ 1 trạng thái bắt đầu.
2. Đánh giá hàm F: Tại mỗi bước, tính giá trị hàm f(n) cho mỗi trạng thái n, là tổng của:
   * g(n): Chi phí để đến trạng thái n từ trạng thái ban đầu.
   * h(n): Ước lượng chi phí từ trạng thái n đến trạng thái mục tiêu.
3. Mở rộng trạng thái: Liệt kê tất cả các trạng thái lân cận có thể đạt được từ trạng thái hiện tại.
4. Chọn lựa tốt nhất: Chọn trạng thái có giá trị hành f tốt nhất.
5. Kiểm tra điều kiện dừng: Dừng khi trạng thái mục tiêu được đạt hoặc không còn trạng thái nào để mở rộng.

**- Giải thuật di truyền (Genetic algorithm)**

Giải thuật di truyền là 1 phương pháp tìm kiếm và tối ưu hóa dựa trên các nguyên tắc của lý thuyết tiến hóa trong sinh học. Cách hoạt động:

1. Khởi tạo quần thể: Bắt đầu bằng việc tạo ra 1 quần thể ban đầu gồm cá thể được mã hóa dưới dạng các gen.
2. Đánh giá: Đánh giá mức độ thích nghi của mỗi cá thể trong quần thể bằng 1 hàm đánh giá.
3. Lựa chọn: Chọn ra 1 số lượng cá thể tốt nhất từ quần thể ban đầu để tạo ra thế hệ mới.
4. Tái tạo: Sử dụng các toán tử như: lai ghép (crossover) và đột biến (mutation) để tạo ra các cá thể mới từ các cá thể được chọn lựa.
5. Tạo ra thế hệ mới: Tạo ra thế hệ mới bằng cách kết hợp các các thể được tái tạo với các cá thể không được chọn lựa.
6. Lặp lại: Lặp lại quá trình từ bước 2 đến bước 5 cho đến khi đạt được điều kiện dừng như: số lượng thế hệ cố định hoặc đạt được giải pháp mong muốn.

**- Giải thuật mô phỏng luyện kim (Simulated Annealing)**

Giải thuật mô phỏng luyện kim là 1 phương pháp tối ưu hóa được lấy cảm hứng từ quá trình luyện kim trong công nghệ vật liệu. Cách hoạt động:

1. Khởi tạo: Bắt đầu từ 1 giải pháp ngẫu nhiên hoặc từ giải pháp khởi tạo.
2. Tạo ra giải pháp lân cận: Di chuyển đến 1 giải pháp lân cận từ giải pháp hiện tại. Các giải pháp lân cận có thể được tạo ra bằng cách thay đổi một phần tử giải pháp hiện tại.
3. Đánh giá mức độ thích nghi: Đánh giá mức độ thích nghi của giải pháp lân cận.
4. Chấp nhận hoặc từ chối: Nếu giải pháp lân cận có mức độ thích nghi tốt hơn giải pháp hiện tại, chấp nhận giải pháp mới. Ngược lại, có một xác suất từ chối giải pháp mới dựa trên 1 hàm lạnh.
5. Giảm nhiệt độ (Annealing): Giảm nhiệt độ của hệ thống theo 1 lịch trình nhất định, điều này giúp giảm dần xác suất của việc chấp nhận giải pháp xấu hơn khi tiến tới giải pháp tối ưu.
6. Lặp lại: Lặp lại quá trình từ bước 3 đến bước 5 cho đến khi đạt được điều kiện dừng như: đạt được số lần lặp cố định hoặc khi nhiệt độ giảm xuống 1 mức nhất định.