1. چهار مرحله کلی برای حل یک مساله را با مثال شهر رومانی شرح دهید؟

حل یک مساله معمولاً به چندین مرحله نیاز دارد. در اینجا چهار مرحله کلی برای حل یک مساله را به همراه مثالی از مساله شهرهای رومانی شرح داده‌ام:

1. تعریف مساله

موضوع مساله: بررسی بهینه‌ترین مسیر بین شهرهای رومانی.

هدف: یافتن مسیر کوتاه‌ترین بین هر دو شهر.

ورودی: نقشه‌ی اتصالات بین شهرها، فاصله‌ها، و اطلاعات مربوط به هر شهر.

2. طراحی راه‌حل

الگوریتم: ایجاد الگوریتم جستجوی بهینه مانند الگوریتم جستجوی گراف، به عنوان مثال.

نقشه‌سازی: تصویر ذهنی یا نمایش نموداری از شهرها و ارتباطات آن‌ها.

استفاده از منابع: تصمیم در مورد استفاده از داده‌ها و منابع مرتبط، مانند فاصله‌ها و اتصالات.

3. پیاده‌سازی

برنامه‌نویسی: نوشتن کد برنامه با استفاده از زبان برنامه‌نویسی موردنظر.

تست و اجرا: اجرای برنامه با داده‌های مختلف و تست صحت عملکرد آن.

4. ارزیابی و بهبود

ارزیابی: بررسی نتایج حاصل از برنامه.

پسخورد: دریافت بازخورد از کاربران یا تغییرات در ورودی‌ها.

بهبود: اقداماتی برای بهبود عملکرد یا کاهش پیچیدگی الگوریتم.

مثال:

فرض کنید می‌خواهید بهینه‌ترین مسیر بین دو شهر در رومانی را پیدا کنید. شهرها به صورت گراف متصل هستند و هر یال وزن فاصله بین دو شهر را نشان می‌دهد. مثلاً، شهرهای A، B، C متصل هستند و وزن یال AB فاصله بین A و B را نشان می‌دهد. در اینجا، مراحل مختلف مسئله ممکن است شامل تعریف گراف شهرها، ایجاد یک الگوریتم جستجوی گراف، نوشتن کد برنامه، تست‌ها و ارزیابی نتایج باشد.

2 . انواع مساله را نام ببرید و شرح مختصری از هر یک با ذکر یک مثال بیان کنید؟

1. مساله جستجو

شرح: در این نوع مساله، هدف یافتن راه‌حل یا وضعیت مشخصی در یک فضا است.

مثال: الگوریتم جستجوی عمق اول یا جستجوی بهینه در گراف‌ها. به عنوان مثال، جستجوی مسیر کوتاه‌ترین بین دو نقطه در یک شبکه جاده‌ای.

2. مساله بهینه‌سازی

شرح: در این نوع مساله، هدف یافتن بهترین راه‌حل از بین مجموعه راه‌حل‌ها است.

مثال: مسئله کمترین مسافت بین چندین شهر با استفاده از الگوریتم بهینه‌سازی گراف.

3. مساله تصمیم

شرح: در این نوع مساله، باید تصمیم بگیرید که آیا یک وضعیت معینی صحیح است یا خیر.

مثال: تشخیص اینکه آیا یک عدد اول است یا خیر.

4. مساله پیش‌بینی

شرح: هدف در این نوع مساله، پیش‌بینی یک ویژگی مشخص در آینده است.

مثال: پیش‌بینی قیمت سهام بر اساس داده‌های مالی و اقتصادی.

5. مساله گراف

شرح: این نوع مساله با ساختار گراف و ارتباطات بین گره‌ها مرتبط است.

مثال: مسئله چگونگی یافتن مسیر کوتاه‌ترین بین دو گره در یک گراف متصل.

6. مساله کلاسیفیکیشن

شرح: هدف در این نوع مساله، انتقال یک مورد به یک دسته یا کلاس مشخص است.

مثال: تشخیص ایمیل‌های هرزنامه با استفاده از الگوریتم‌های یادگیری ماشین.

7. مساله خوشه‌بندی

شرح: در این نوع مساله، موارد مشابه در گروه‌های جداگانه قرار می‌گیرند.

مثال: خوشه‌بندی مشتریان بر اساس رفتار خرید و ترجیحات آن‌ها.

8. مساله بهبود

شرح: در این نوع مساله، هدف بهبود یک وضعیت مشخص است.

مثال: بهبود عملکرد یک الگوریتم با تغییرات در ساختار یا پارامترهای آن.

3 . مسئله 8 وزیر را با دو روش فرموله سازی کنید(مثال n وزیر را طوری در صفحه شطرنج بگذارید که همدیگر را تهدید نکنند)؟

4 . جستجوی درختی را با ذکر یک مثال شرح دهید؟

جستجوی درختی یکی از الگوریتم‌های جستجو در گراف‌ها است که بر اساس ساختار درختی اطلاعات را جستجو می‌کند. این الگوریتم معمولاً برای حل مسائلی استفاده می‌شود که به صورت درختی قابل مدل‌سازی باشند، مانند مسائل جستجو در فضای وضعیت‌ها یا تصمیم‌گیری‌های درختی.

مثال: جستجوی درختی در مسئله یافتن مسیر کوتاه:

فرض کنید یک گراف وزن‌دار داریم که نمایانگر شهرها و اتصالات بین آن‌ها است. هدف ما یافتن مسیر کوتاه‌ترین بین دو شهر مشخص است. از الگوریتم جستجوی درختی برای حل این مسئله استفاده می‌کنیم.

5 . فضای حالت و Fringe را تعریف کنید؟

فضای حالت، مجموعه‌ای از تمام وضعیت‌های ممکن یک سیستم یا مسئله را نشان می‌دهد. هر وضعیت ممکن یک نقطه در فضای حالت است. در مسائل جستجو و هوش مصنوعی، مفهوم فضای حالت برای مدل کردن وضعیت‌ها و تغییرات آن‌ها استفاده می‌شود. همچنین، مسائل به صورت گراف معمولاً با گره‌هایی که وضعیت‌ها را نمایان می‌دهند، در فضای حالت مدل می‌شوند.

Fringe یا لبه، مجموعه از گره‌های مرزی یا گره‌هایی است که در مرحله جاری اجرای الگوریتم جستجو به آن‌ها رسیده است. این گره‌ها ممکن است شامل گره‌های مرزی بین حالت‌های باز و حالت‌های بسته شده باشند. در اصطلاحات الگوریتم‌های جستجو، Fringe به عنوان محدوده جستجو نشان داده می‌شود و در هر مرحله، گره‌های Fringe بر اساس استراتژی جستجو انتخاب می‌شوند تا بررسی و یا گام بعدی جستجو انجام شود.

به عنوان مثال، در الگوریتم جستجوی سپسیس، Fringe می‌تواند به عنوان مجموعه‌ای از گره‌های مرزی باشد که به تازگی بررسی شده‌اند و در انتظار بررسی بیشتر هستند.

6 . جستجوی ناآگاهانه را تعریف کنید و انواع آن را نام ببرید؟

جستجوی ناآگاهانه یا جستجوی کور، الگوریتم‌هایی هستند که بدون در نظر گرفتن اطلاعات خاص درباره مسئله، با آزمون و خطا به جستجو می‌پردازند. این الگوریتم‌ها بر اساس قوانین و حدس‌های کلی عمل می‌کنند و از دانش خاصی درباره ساختار مسئله استفاده نمی‌کنند. این الگوریتم‌ها بیشتر برای مسائلی با فضای حالت بزرگ یا بدون اطلاعات خاص درباره محیط استفاده می‌شوند.

1. جستجوی سپسیس
2. جستجوی عمق اول
3. جستجوی یکپارچه
4. جستجوی عرضه اول

7 . الگوریتمی که از لحاظ زمانی از مرتبه جستجوی اول سطح است ولی از لحاظ پیچیدگی حافظه از مرتبه جستجوی اول عمق می باشد کدام است ،شرح دهید؟

جستجوی یکپارچه

به عنوان یک الگوریتم بهینه در جستجوهای یکپارچه شناخته می‌شود، اما همواره نیازمند UCS حافظه بسیار زیادی برای ذخیره اطلاعات درخت جستجو است. این در حالی است که DFS با مصرف حافظه کمتر عمل می‌کند اما ممکن است به جواب بهینه نرسد.

8 . کارایی انواع جستجوهای نا آگاهانه را بر حسب چهار پارامتر کامل بودن ،بهینگی ، پیچیدگی زمانی و فضایی بیان کنید؟

جستجوی BFS

کامل بودن: الگوریتم کامل است و در صورت وجود جواب، جواب را پیدا می‌کند.

بهینگی: بهینه نیست، زیرا ممکن است گره‌ها هزینه‌های مختلفی داشته باشند و ترتیب افزوده شدن به Fringe اهمیت بهینگی را نداشته باشد.

پیچیدگی زمانی: برای گراف‌های محدود، پیچیدگی زمانی خوب است (از مرتبه $O(b^d)$ که b تعداد شاخه‌ها و d عمق حلقه را نشان می‌دهد).

پیچیدگی فضایی: پیچیدگی فضایی نسبتاً بالاست، زیرا باید تمام گره‌های یک سطح را در حافظه نگه‌داری کند (از مرتبه $O(b^d)$)

جستجوی عمق اول

کامل بودن: الگوریتم کامل نیست، زیرا ممکن است به حالت گیر بیفتد (برای گراف‌های بی‌نهایت یا به تعداد زیادی حلقه).

بهینگی: بهینه نیست، زیرا ممکن است به حلقه بخورد و همچنین مسیرها را به ترتیب پیش نگاه نمی‌کند.

پیچیدگی زمانی: برای گراف‌های محدود، پیچیدگی زمانی خوب است (از مرتبه $O(b^m)$ که b تعداد شاخه‌ها و m عمق حداکثر را نشان می‌دهد).

پیچیدگی فضایی: پیچیدگی فضایی متناسب با عمق حداکثر مسیر است (از مرتبه $O(b^m)$)

جستجوی یکپارچه

کامل بودن: الگوریتم کامل است، زیرا در پایان الگوریتم تمام گره‌ها بررسی می‌شوند.

بهینگی: بهینه است، زیرا هزینه حالت‌ها به ترتیب صعودی ارزیابی می‌شود.

پیچیدگی زمانی: اگر هزینه‌ها یکسان باشند، پیچیدگی زمانی بسیار بالا خواهد بود (از مرتبه $O(b^{1+[c^*/\varepsilon]})$ که c^* هزینه بهینه و ε کمترین هزینه مثبت است)

پیچیدگی فضایی: پیچیدگی فضایی متناسب با تعداد گره‌ها است (از مرتبه $O(b^{1+[c^*/\varepsilon]})$.