>_ BÀI 3 · PROMPT ENGINEERING

Kỹ Thuật Thiết Kế Prompt Chuyên Sâu

Thực nghiệm so sánh sự biến thiên chất lượng đầu ra giữa prompt cơ bản, cải tiến, nâng cao trong các tác vụ phân tích thuật toán ALT và tối ưu hóa Heuristic.

Tác vụ 1: Tóm tắt Bài báo Khoa học

Tác vụ bóc tách thuật toán ALT (Goldberg et al.)

CƠ BẢN

Prompt:

```
Tóm tắt bài báo "Computing the Shortest Path: A* Search Meets Graph Theory" của
Andrew V. Goldberg.
```

Kết quả đầu ra:

Bài báo này của Microsoft Research nghiên cứu về việc tính toán đường đi ngắn nhất trên đồ thị lớn bằng thuật toán A*. Nhóm nghiên cứu đề xuất một thuật toán cải tiến mới giúp tìm đường nhanh hơn và hiệu quả hơn so với thuật toán cũ.

Bản chất kỹ thuật: Không cung cấp vai trò, mục tiêu cụ thể hay ràng buộc ngữ cảnh. AI đưa ra kết quả rất chung chung, mang tính khái quát bề nổi, thiếu hoàn toàn thông tin về giải thuật cốt lõi (ALT), mốc (Landmarks) hay bất đẳng thức tam giác.

CẢI TIẾN

Prompt:

```
Hãy tóm tắt nội dung cốt lõi, phương pháp nghiên cứu và kết luận của bài báo
"Computing the Shortest Path: A* Search Meets Graph Theory" của Andrew V. Goldberg.
Phân tích cụ thể cách thuật toán A* kết hợp với Graph Theory.
```

Kết quả đầu ra:

Bài báo đề xuất thuật toán ALT - kết hợp thuật toán tìm đường A* với lý thuyết đồ thị thông qua các điểm mốc (Landmarks) và bất đẳng thức tam giác để tính toán heuristic tốt hơn.
Phương pháp nghiên cứu gồm tiền xử lý đồ thị để tính khoảng cách đến các Landmarks, sau đó dùng Heuristic để truy vấn nhanh. Kết luận chỉ ra tốc độ truy vấn tăng đáng kể.

Bản chất kỹ thuật: Cấu trúc đầu ra tốt hơn nhờ yêu cầu rõ ràng các mục tóm tắt. Tuy nhiên, phần giải thích cơ chế kỹ thuật toán học (bất đẳng thức tam giác hoạt động như thế nào để tối ưu Heuristic) vẫn còn khá mơ hồ và chưa sâu sắc.

NÂNG CAO

Prompt:

```
Đóng vai một Giáo sư Khoa học Máy tính chuyên giảng dạy về Cấu trúc Dữ liệu và Giải
thuật. Hãy tóm tắt bài báo "Computing the Shortest Path: A* Search Meets Graph
Theory" của Andrew V. Goldberg et al.

Thực hiện theo các bước tư duy sau:
1. Xác định nút thắt (bottleneck) của thuật toán Dijkstra và A* truyền thống trên đồ
thị cực lớn.
2. Giải thích cơ chế cốt lõi của thuật toán ALT (A* search, Landmarks, Triangle
inequality). Bất đẳng thức tam giác đã được sử dụng để tối ưu hóa hàm heuristic như
thế nào?
3. Đánh giá sự đánh đổi (trade-off) giữa chi phí tiền xử lý (preprocessing) và tốc
độ truy vấn
```

Kết quả đầu ra:

• Bottleneck: Thuật toán Dijkstra duyệt đồ thị mù quáng theo mọi hướng nên tốn bộ nhớ. A* truyền thống khắc phục bằng hàm heuristic định hướng, nhưng trên đồ thị khổng lồ (như bản đồ đường bộ quốc gia), việc thiết kế một heuristic đủ mạnh mà vẫn bảo đảm tính Admissible là vô cùng khó khăn.

• Cơ chế thuật toán ALT (Landmarks & Triangle Inequality):
- Chọn trước một số đỉnh mốc L (Landmarks) trên đồ thị.
- Tính toán và lưu trữ trước khoảng cách d(u, L) từ mọi đỉnh u đến L.
- Trong pha truy vấn tìm đường từ u đến đích v, áp dụng bất đẳng thức tam giác: d(u, v) >= d(u, L) - d(v, L) và d(u, v) >= d(v, L) - d(u, L). Từ đó suy ra heuristic tối ưu: h(u) = max_{L} |d(u, L) - d(v, L)|.
- Heuristic này đảm bảo cả hai thuộc tính Admissible và Consistent, giúp thu hẹp đáng kể số lượng node cần duyệt mà không làm mất tính tối ưu của đường đi.

• Phân tích Trade-off:
- Pha tiền xử lý (Preprocessing): Đòi hỏi chi phí tính toán lớn và không gian lưu trữ bổ sung để chứa ma trận khoảng cách đến các Landmarks.
- Pha truy vấn (Query): Tốc độ tìm kiếm được tối ưu hóa cực đại (nhanh hơn hàng chục đến hàng trăm lần), phù hợp cho các tác vụ thời gian thực như hệ thống định vị GPS.

Bản chất kỹ thuật: Ứng dụng thành thạo **Role Prompting**, **Chain-of-Thought (Tư duy từng bước)** và **Ràng buộc định dạng**. Ép mô hình AI lọc các trọng số từ vựng chuyên ngành sâu sắc, đưa ra lý giải toán học chính xác và logic phản biện chặt chẽ.

Tác vụ 2: Giải thích Khái niệm Phức tạp

Tối ưu hóa heuristic (Admissible & Consistent)

CƠ BẢN

Prompt:

```
Giải thích khái niệm Tối ưu hóa heuristic.
```

Kết quả đầu ra:

Tối ưu hóa heuristic là phương pháp thiết kế các quy tắc hoặc hàm đánh giá để giúp thuật toán tìm kiếm giải pháp nhanh hơn. Nó không bảo đảm tìm ra giải pháp tốt nhất nhưng giúp tiết kiệm thời gian tính toán.

Bản chất kỹ thuật: Đưa ra định nghĩa từ điển học thuật khô khan, thiếu ví dụ thực tế nên rất khó tiếp thu đối với sinh viên mới bắt đầu học giải thuật.

CẢI TIẾN

Prompt:

```
Giải thích khái niệm Tối ưu hóa heuristic trong thuật toán tìm đường (như A*). Lấy một ví dụ minh họa để làm rõ tại sao nó lại quan trọng.
```

Kết quả đầu ra:

Trong thuật toán tìm đường A*, tối ưu hóa heuristic là việc thiết kế hàm h(n) để ước lượng khoảng cách từ node hiện tại đến đích. Ví dụ, trên bản đồ thành phố, ta có thể dùng khoảng cách đường chim bay làm Heuristic để robot hướng về phía đích. Việc này giúp thuật toán không phải duyệt các hướng ngược lại, tiết kiệm thời gian xử lý.

Bản chất kỹ thuật: Thêm ví dụ minh họa trực quan giúp dễ hiểu hơn rất nhiều. Tuy nhiên, câu trả lời vẫn chưa đi sâu vào cơ chế toán học của sự tối ưu (lý giải tính Admissible).

NÂNG CAO

Prompt:

```
Bạn là một Kỹ sư Hệ thống dày dạn kinh nghiệm đang hướng dẫn một sinh viên năm nhất trường công nghệ. Hãy giải thích khái niệm "Tối ưu hóa heuristic" (Heuristic optimization) trong thuật toán A*.

Yêu cầu:
- Sử dụng phương pháp loại suy (analogy): So sánh hàm heuristic với hệ thống định vị hoặc trực giác của con người khi đi đường.
- Phải giải thích rõ hai thuộc tính cốt lõi để một heuristic được coi là "tối ưu": Admissible (không ước lượng vượt quá chi phí thực) và Consistent (tính nhất quán).
- Kết luận bằng việc tại sao tối ưu hóa heuristic lại quyết định ranh giới giữa một đoạn code chạy mất 10 giây và 10 mili-giây.
```

Kết quả đầu ra:

Hãy tưởng tượng em đang đi bộ trong một mê cung để đến một tòa nhà cao tầng. Trực giác của em luôn bảo rằng: "Hãy bước về hướng tòa nhà đó". Hàm Heuristic trong A* chính là "trực giác số" đó, giúp thuật toán định hướng thay vì đi mò mẫm khắp nơi.

Để trực giác này không dẫn em đi sai đường, nó bắt buộc phải thỏa mãn 2 điều kiện:
1. Tính Admissible (Chấp nhận được): Hàm ước lượng h(n) tuyệt đối không được lớn hơn chi phí đi thực tế. Giống như ước lượng khoảng cách đường chim bay – nó luôn ngắn hơn hoặc bằng quãng đường đi thực tế qua các ngõ ngách. Nếu ước lượng quá lớn, thuật toán sẽ bỏ qua con đường ngắn nhất thật sự.
2. Tính Consistent (Nhất quán/Đơn điệu): Chi phí ước lượng từ node hiện tại đến đích không được lớn hơn chi phí đi sang node liền kề cộng với chi phí ước lượng từ node liền kề đó đến đích. Điều này đảm bảo rằng khi thuật toán duyệt qua một node, đường đi đến node đó đã là ngắn nhất, không cần tính toán lại.

Kết luận: Nếu không tối ưu hóa Heuristic (hoặc thiết kế h(n) = 0), A* sẽ thoái hóa thành Dijkstra tìm kiếm mù quáng, làm chương trình xử lý mất 10 giây trên bản đồ lớn. Nhưng với một Heuristic tối ưu, không gian tìm kiếm được thu hẹp tối đa, ép thời gian chạy xuống dưới 10 mili-giây.

Bản chất kỹ thuật: Kết hợp **Role Prompting**, **Phương pháp loại suy (Analogy)** và **Ràng buộc nội dung khắt khe**. Giúp chuyển tải kiến thức toán học trừu tượng thành trực quan sinh động mà không mất đi tính chính xác.

Tác vụ 3: Thiết kế Bộ Câu hỏi Ôn tập

Tạo câu hỏi phân hóa theo thang đo nhận thức Bloom

CƠ BẢN

Prompt:

```
Tạo 5 câu hỏi trắc nghiệm về thuật toán A* và kỹ thuật Landmarks.
```

Kết quả đầu ra:

1. Thuật toán A* dùng để làm gì?
2. Hàm heuristic trong A* ký hiệu là gì?
3. Landmarks là gì?
4. Bất đẳng thức tam giác là gì?
5. Thuật toán ALT viết tắt của cụm từ nào?

Bản chất kỹ thuật: Các câu hỏi cực kỳ hời hợt, chỉ dừng lại ở mức độ học thuộc lòng định nghĩa thô sơ. Không có đáp án và giải thích đi kèm.

CẢI TIẾN

Prompt:

```
Dựa trên bài báo của Andrew V. Goldberg về thuật toán ALT, hãy tạo một bài kiểm tra gồm 5 câu hỏi trắc nghiệm khách quan có 4 đáp án (A, B, C, D). Yêu cầu cung cấp đáp án đúng và giải thích ngắn gọn ở cuối.
```

Kết quả đầu ra:

[Bài kiểm tra 5 câu hỏi về thuật toán ALT]
(Ví dụ câu hỏi):
Câu 1: Hàm Heuristic trong thuật toán ALT được tính toán dựa trên:
A) Dijkstra
B) Landmarks và bất đẳng thức tam giác
C) Trực giác con người
D) Trí tuệ nhân tạo học máy
=> Đáp án đúng: B. Giải thích: ALT kết hợp A*, Landmarks và Triangle inequality.

Bản chất kỹ thuật: Đã có định dạng đầu ra rõ ràng, tuy nhiên các câu hỏi chỉ dừng lại ở mức độ nhận biết thông thường do thiếu một khung đánh giá sư phạm định hình.

NÂNG CAO

Prompt:

```
Đóng vai chuyên gia ra đề thi môn Phân tích Thiết kế Thuật toán. Dựa vào nội dung
thuật toán ALT (A*, Landmarks, Triangle inequality) của Andrew V. Goldberg, hãy tạo
một bài trắc nghiệm 5 câu.

Áp dụng khung phân loại nhận thức Bloom:
- 2 câu ở mức độ Nhớ (Recall): Hỏi về định nghĩa và thành phần cốt lõi.
- 2 câu ở mức độ Hiểu (Understand): Đòi hỏi suy luận về vai trò của bất đẳng thức
tam giác trong ALT.
- 1 câu ở mức độ Vận dụng (Apply): Đưa ra một tình huống giả định về số lượng
Landmarks hoặc chi phí tiền xử lý để đánh giá trade-off.

Định dạng xuất ra:
```

Kết quả đầu ra:

[ĐỀ THI ĐÁNH GIÁ CHUYÊN SÂU - THUẬT TOÁN ALT]

--- CẤP ĐỘ 1: NHỚ (RECALL) ---

Câu 1: Trong thuật toán ALT, thành phần dữ liệu nào được tính toán trước ở pha Preprocessing?
A) Quỹ đạo di chuyển cuối cùng của mọi cặp đỉnh.
B) Khoảng cách từ mọi đỉnh đến tập hợp các điểm mốc (Landmarks) đã chọn.
C) Trọng số thay đổi của các cạnh trong môi trường động.
D) Toàn bộ hàm heuristic của mọi đỉnh trên đồ thị.
=> Đáp án đúng: B
=> Giải thích: Pha tiền xử lý tính toán và lưu trữ khoảng cách d(u, L) từ mọi đỉnh u đến mọi điểm mốc L để làm cơ sở tính Heuristic nhanh trong pha truy vấn.

Câu 2: Ký hiệu của thuật toán ALT đại diện cho sự kết hợp của những yếu tố nào?
A) Algorithm, Link-analysis, Traversal.
B) A* Search, Landmarks, Triangle Inequality.
C) Asymptotic, Limit-bound, Triangulation.
D) Admissible, Landmark-distance, Tree-structure.
=> Đáp án đúng: B
=> Giải thích: ALT viết tắt của A* search, Landmarks, và Triangle inequality (bất đẳng thức tam giác).

--- CẤP ĐỘ 2: HIỂU (UNDERSTAND) ---

Câu 3: Bất đẳng thức tam giác nào sau đây được sử dụng để xây dựng một heuristic dưới hạn dưới (lower bound) hợp lệ cho khoảng cách d(u, v) trong thuật toán ALT?
A) d(u, v) <= d(u, L) + d(v, L)
B) d(u, v) >= |d(u, L) - d(v, L)|
C) d(u, v) >= d(u, L) + d(L, v)
D) d(u, v) <= |d(u, L) - d(v, L)|
=> Đáp án đúng: B
=> Giải thích: Theo bất đẳng thức tam giác, ta luôn có d(u, L) <= d(u, v) + d(v, L) => d(u, v) >= d(u, L) - d(v, L). Tương tự d(u, v) >= d(v, L) - d(u, L). Do đó h(u) = max |d(u, L) - d(v, L)| là một giới hạn dưới hợp lệ (Admissible).

Câu 4: Tại sao hàm Heuristic h(u) = max_{L} |d(u, L) - d(v, L)| trong thuật toán ALT luôn đảm bảo tính nhất quán (Consistent)?
A) Vì nó dựa trên khoảng cách Euclid trên mặt phẳng.

B) Vì bất đẳng thức tam giác tự đóng vai trò là một điều kiện ràng buộc biên dịch chuyển chi phí đơn điệu giữa các node kề nhau.
C) Vì số lượng Landmarks luôn lớn hơn số lượng đỉnh của đồ thị.
D) Vì khoảng cách đến Landmarks được cập nhật liên tục trong pha truy vấn.
=> Đáp án đúng: B
=> Giải thích: Tính chất bất đẳng thức tam giác bảo đảm rằng với mọi cạnh (u, w), hiệu khoảng cách tới Landmark thay đổi không vượt quá trọng số cạnh w(u, w), thỏa mãn điều kiện nhất quán h(u) <= w(u, w) + h(w).

--- CẤP ĐỘ 3: VẬN DỤNG (APPLY) ---

Câu 5: Giả sử em thiết lập một hệ thống dẫn đường bản đồ GPS với 1.000.000 đỉnh. Nếu em tăng số lượng Landmarks được chọn từ 10 điểm lên 100 điểm, hệ quả nào sau đây sẽ xảy ra đối với hệ thống?
A) Thời gian tiền xử lý giảm, tốc độ truy vấn tăng, bộ nhớ RAM yêu cầu giảm.
B) Thời gian tiền xử lý tăng, tốc độ truy vấn tăng (duyệt ít node hơn), bộ nhớ RAM yêu cầu tăng gấp 10 lần.
C) Không ảnh hưởng đến bộ nhớ RAM nhưng làm tăng độ phức tạp thời gian truy vấn lên gấp 10 lần.
D) Thuật toán sẽ mất đi tính tối ưu do hàm Heuristic đánh giá quá cao khoảng cách thực tế.
=> Đáp án đúng: B
=> Giải thích: Tăng số lượng Landmarks giúp ước lượng Heuristic sát với khoảng cách thực tế hơn, do đó thu hẹp không gian tìm kiếm làm pha truy vấn nhanh hơn. Tuy nhiên, chi phí tiền xử lý tăng lên và bộ nhớ lưu trữ khoảng cách d(u, L) tăng tuyến tính theo số lượng Landmark (từ 10 x 1M lên 100 x 1M bản ghi), thể hiện rõ sự đánh đổi Space-Time Tradeoff.

Bản chất kỹ thuật: Tích hợp **Khung phân loại nhận thức Bloom** vào prompt để phân tầng câu hỏi khoa học từ cơ bản đến nâng cao. Định dạng đầu ra được kiểm soát chi tiết giúp bộ đề thi có tính ứng dụng sư phạm xuất sắc.