دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گزارش پروژه دوم

مبانی و کاربردهای هوش مصنوعی

نگارش

آرش حاجی صفی - ۹۶۳۱۰۱۹

بهمن ۱۳۹۹

فرموله‌سازی مسئله:

تمامی کلاس‌ها و متدهایی که مربوط به حل مسئله هست (فارغ از نحوه پیاده‌سازی الگوریتم جستجو)، در فایل data_structure.py قرار داده شده اند و در فایل sudoku_solver.py که مسئله حل می‌شود، import شده اند. در ادامه به توضیح این موارد که برای فرموله‌سازی مسئله به کار رفته‌اند می‌پردازم:

فرموله‌سازی متغیرها

کلاسی را به اسم Cell پیاده‌سازی نموده‌ام که دارای یک شماره، یک رنگ، مختصات آن در جدول، یک دامنه از رنگ‌ها و دامنه‌ای از اعداد، یک لیست از مختصات Cellهای دیگری که با آنها در محدودیت برای شماره ارتباط دارد، و لیست دیگری از مختصات Cellهایی که با آنها در محدودیت رنگ ارتباط دارد را شامل می‌شود. در اصل هر شیء از جنس این کلاس، یک متغیر در مسئله اصلی است و مسئله ما این است که باید فیلدهای عدد و رنگ تمامی Cellها را طوری مقداردهی کنیم که محدودیت‌ها برآورده شوند.

فرموله‌سازی دامنه‌ها

هر Cell (که یک متغیر در مسئله ما است) دارای یک لیست است که دامنه عددی آنرا نگه می‌دارد (لیستی از اعداد ممکن برای مقداردهی فیلد number آن Cell) و دارای لیست دیگری است که دامنه رنگ‌های ممکن برای مقدار دهی فیلد color این Cell را نگه می‌دارد (لیستی از اعداد که هر عدد به صورت string ذخیره شده است).

فرموله‌سازی محدودیت‌ها

برای محدودیت، کلاس Constraint را تعریف نموده‌ام که هر شیء از این کلاس دارای دو فیلد i و j است که در اصل مختصات هستند و یک خانه از جدول را مشخص می‌کنند. هر شیء از جنس کلاس Cell (متغیر ما) دارای ۲ لیست از اشیاء از جنس Constraint است. در لیست اول که number_constraints نام دارد، مختصات تمامی سلول‌هایی که در جدول مسئله مورد نظر با این سلول مورد نظر در محدودیت عدد ارتباط دارند، در قالب اشیاء از جنس Constraint ذخیره شده‌اند. یعنی مختصات تمامی سلول‌هایی که با این سلول در یک سطر و یا در یک ستون قرار گرفته‌اند در این لیست قرار می‌گیرند.

در لیست دوم داخل کلاس Cell که color_constraints نام دارد، مختصات تمامی سلول‌هایی که در جدول مسئله مورد نظر با این سلول مورد نظر در محدودیت رنگ ارتباط دارند، در قالب اشیاء از جنس Constraint ذخیره می‌شود. یعنی مختصات تمامی سلول‌های مجاور سلول مورد نظر در این لیست قرار دارند.

توضیح کد نوشته شده برای حل مسئله:

یک کلاس به اسم **کلاس CSP** تعریف نموده‌ام که شامل لیستی دو بعدی از Cellها که متغیرهای مسئله ما هستند می‌شود. این لیست دو بعدی از سلول‌ها در اصل همان جدول مسئله مورد نظر ما هستند و هر در هرجای این لیست معادل متغیر مورد نظر در خانه مربوطه در جدول است. برای حل مسئله با مقداردهی اولیه این سلول‌ها از روی فایل تست کیس و دادن لیست دوبعدی مورد نظر از آنها به این کلاس، شیئی از این جنس را می‌سازیم. در constructor این شیء، برای هر سلول در لیست، مقداردهی اولیه

دامنه‌ها (هم دامنه اعداد و هم دامنه رنگ‌ها) صورت می‌گیرد و لیست محدودیت‌ها برای آن سلول نیز پر می‌شود. سپس تابع backtrack_search() روی این شیء csp صدا زده می‌شود که این تابع پیاده‌سازی الگوریتم backtrack به صورت بازگشتی است.

در هربار اجرای این تابع، ابتدا بررسی می‌شود که همه متغیرها فیلدهای عدد و رنگشان مقداردهی شده‌است یا نه؛ اگر اینطور بود که الگوریتم به اتمام رسیده و شیء csp حاصل که تمامی مقادیر در آن به متغیرها assign شده‌اند برگردانده می‌شود. در غیر این صورت، ابتدا یک کپی از این شیء csp گرفته می‌شود، سپس با فراخوانی تابع next_var روی csp، با هیوریستیک mrv سلول بعدی برای مقداردهی انتخاب می‌شود؛ به این صورت که ابتدا با استفاده از هیوریستیک mrv، بر اساس دامنه رنگ‌ها متغرهایی که کمترین تعداد المنت باقی‌مانده از رنگ در دامنه را دارند انتخاب می‌شوند، به همین صورت با همین هیوریستیک، متغیرهایی که کمترین تعداد المنت باقی‌مانده از عدد را دارند انتخاب می‌شوند؛ سپس دامنه هر کدام از این دو دسته که کوچکتر باشد (دامنه رنگ یا عدد)، آن گروه متغیر انتخاب شده و اگر بیش از یک متغیر توسط mrv انتخاب شده بود (یعنی چندین متغیر حداقل دامنه را داشتند)، با استفاده از هیوریستیک درجه آن سلولی را که درجه بالاتری دارد انتخاب می‌کنیم و مقداری را به صورت تصادفی برای پارامتر مورد نظر (پارامتر رنگ یا عدد که mrv آنرا انتخاب کرده) از روی دامنه آن پارامتر در سلول مورد نظر انتخاب می‌کنیم و به آن assign می‌کنیم. سپس با فراخوانی تابع forward_check روی csp استدلال‌های مورد نظر را انجام داده و دامنه سلول‌هایی که با این سلول در محدودیتی در ارتباط بوده‌اند را آپدیت می‌کنیم و دوباره به صورت بازگشتی backtrack_search را روی csp حاصل فراخوانی می‌کنیم، اگر که failure برگردانده شود، یعنی backtrack صورت گرفته و با فراخوانی تابع restore_inferences استدلال‌های انجام شده برای دامنه‌ها را حذف می‌کنیم و به حالت قبلی برمی‌گردیم و مقداری جدید را از دامنه سلول انتخاب شده بر می‌داریم و برای پارامتر مورد نظرش assign می‌کنیم. نهایتاً اگر برای سلولی همه‌ی مقادیر دامنه مورد نظرش را امتحان کردیم و در هیچ حالتی به جواب نرسیدیم، failure برگردانده می‌شود و برای مسئله مورد نظر جوابی وجود ندارد. در صورت وجود جواب نیز csp نتیجه که همه مقادیر رنگ و عدد سلول‌ها در آن مقداردهی شده‌اند برگردانده می‌شود.

نتیجه اجرای الگوریتم روی مثال صورت پروژه: