Universidade Federal do Rio Grande do Norte

Escola Agrícola de Jundiaí

Análise e Desenvolvimento de Sistemas

TAD0025 - Inteligência Computacional

Esse projeto corresponde a nota da segunda unidade e pode ser desenvolvido em dupla ou individualmente. O aluno deve escolher **um** entre os três projetos apresentados abaixo. Para o projeto 1, submeta somente o artigo desenvolvido no formato PDF. Para os projetos 2 e 3, submeta um arquivo compactado com o código-fonte do projeto incluindo o readme.txt (**não inclua pastas, executáveis e nem arquivos de configuração do ambiente de desenvolvimento**). O readme do projeto 2 deve apresentar a base de conhecimento e informações necessárias para configurar o sistema, enquanto que o do projeto 3 deve apresentar as tabelas de resultados dos experimentos, explicar a codificação das soluções, a função de aptidão com exemplo, os parâmetros selecionados e a melhor solução encontrada.

Projeto 1 - Aplicação de Sistemas Especialistas

Desenvolva um sistema baseado em conhecimento ou sistema especialista, baseado em regras.

- Você é livre para escolher a área do domínio de aplicação com preferência para a área agrícola.
- A aplicação não pode ser trivial, apresentando uma hierarquia das variáveis e uma quantidade razoável de regras (ao menos 20).
- O sistema deve ser descrito no formato de um artigo (manual-de-elaboracao-do-artigo.pdf (ufpb.br)) seguindo o modelo da SBC (Templates para Artigos e Capitulos de Livros (sbc.org.br)). Siga as sugestões para a escrita:

1. Resumo: deve descrever o contexto, o objetivo, os resultados e conclusões de modo sintetizado. Não é necessário o resumo em inglês.
2. Introdução: deve descrever o problema a ser resolvido, a motivação para o trabalho, a área de aplicação, limitações, justificar o desenvolvimento e deixar claro os objetivos.
3. Referencial teórico: deve discutir as possíveis abordagens para o desenvolvimento da aplicação (descrever ao menos o Expert Sinta e o Prolog) fazendo uma comparação das ferramentas disponíveis e justificar a escolha de uma delas. Também deve incluir a descrição de soluções existentes para o problema, caso elas existam.
4. Metodologia: deve descrever como o trabalho foi desenvolvido indicando como o conhecimento foi obtido, sua representação incluindo variáveis e fatos envolvidos, e o mecanismo de raciocínio empregado.
5. Resultados e discussões: deve analisar a solução do problema conforme os objetivos traçados. Deve incluir os problemas encontrados ao longo do desenvolvimento como também problemas na solução.
6. Considerações finais: deve responder se os objetivos foram alcançados, se o problema foi solucionado, comentar limitações, recomendações e trabalhos futuros.
7. Anexo: deve apresentar as regras do sistema.

Projeto 2 - Desenvolvimento de uma Ferramenta para Criação Sistemas Especialistas

Desenvolva uma ferramenta para criação de sistemas especialistas baseados em regras.

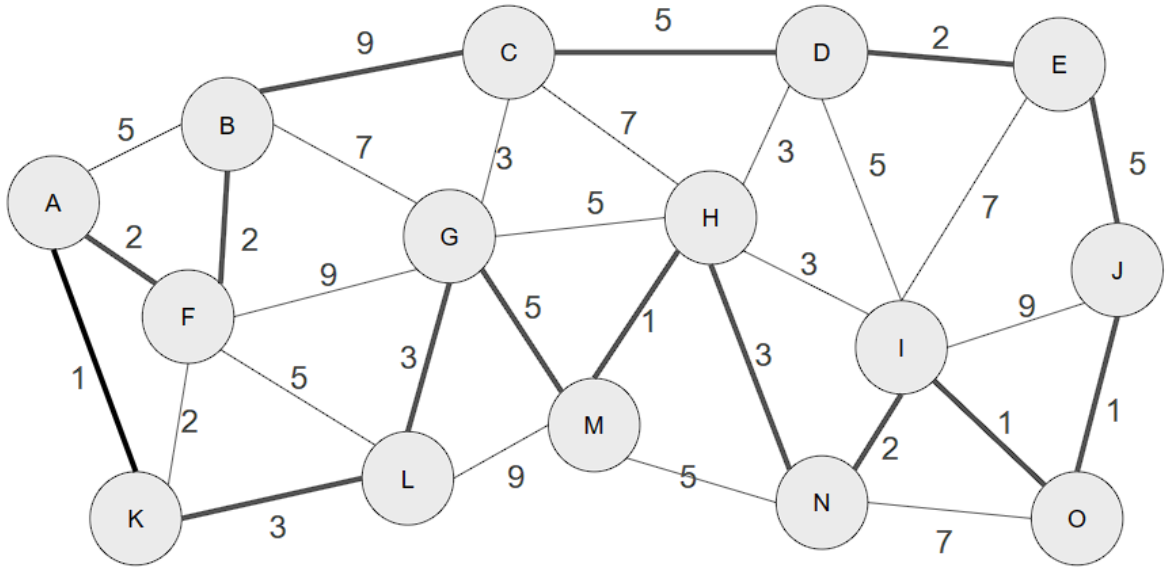
- Você é livre para escolher a linguagem de programação;
- A implementação deve destacar os módulos de base de conhecimento, raciocínio e interface do usuário;
- A interface pode ser desenvolvida em modo textual;
- A ferramenta deve apresentar as seguintes funcionalidades:

1. Customização das regras num arquivo separado da implementação. Esse arquivo deve conter as regras no formato SE <antecedente> ENTÃO <consequente> seguindo a sintaxe de regras de produção vista em aula. Esse arquivo deve ser legível, isto é, o conteúdo pode ser entendido por uma pessoa com pouco conhecimento em linguagens de programação.
2. Além disso, o arquivo pode iniciar com uma seção de descrição das variáveis e valores que elas podem assumir, incluindo a indicação se uma variável corresponde a uma entrada ou objetivo do sistema. O sistema deve suportar variáveis quantitativas e qualitativas, e deve permitir a associação entre perguntas e variáveis;
3. Estratégias para lidar com conflitos entre as regras. Essas estratégias podem ser configuradas no arquivo da base de conhecimento;
4. Mecanismo de raciocínio progressivo usando regras de produção. Utilize o algoritmo de raciocínio progressivo visto em aula;
5. Mecanismo para explicação das conclusões exibindo a ativação das regras;
6. A execução mínima do programa deve apresentar as perguntas associadas a cada variável, culminando numa resposta que pode ser uma conclusão sobre alguma variável objetivo ou a notificação de que não foi possível chegar a alguma conclusão.
7. A execução do programa deve solicitar um arquivo da base de conhecimento para sua execução.

8. Inclua um readme.txt no projeto explicando o formato do arquivo da base de conhecimento.
9. Inclua um exemplo de sistema especialista para o seu programa, por exemplo, sistema de diagnóstico para COVID-19.

Projeto 3 - Desenvolvimento de Algoritmo Genético

Desenvolva um algoritmo genético para resolver o problema do caixeiro viajante considerando as cidades e suas conexões apresentadas no diagrama abaixo:



O problema do caixeiro viajante consiste em encontrar um percurso que parte de uma cidade X qualquer e passa por todas as outras até retornar a X. O percurso deve passar obrigatoriamente por todas as cidades uma única vez e o objetivo é encontrar o percurso mais curto. A distância de um percurso é medida como a soma de todas as conexões que fazem parte do percurso. Por exemplo, a imagem acima destaca o percurso <A, F, B, C, D, E, J, O, I, N, H, M, G, L, K> em que a distância é 2+2+9+5+2+5+1+1+2+3+1+5+3+3+1 = 45. Além disso, assuma que todas as cidades estão conectadas considerando o peso 10 para as conexões não listadas no diagrama, por exemplo, a conexão de A para C não foi listada e nesse caso possui o peso 10.

O trabalho deve encontrar o percurso de menor distância entre todos os percursos possíveis. Para isso, siga os seguintes passos:

1. Escolha uma representação das soluções para serem usadas no algoritmo genético;

2. Determine a função de custo;
3. Modifique o algoritmo genético apresentado em aula para suportar a representação, função de custo escolhida (**não utilize outras implementações ou bibliotecas, pois não serão aceitas**);
4. Modifique os adicione operadores genéticos adequados à representação escolhida;
5. Determine um conjunto de valores razoáveis para cada parâmetro do algoritmo genético, por exemplo:
    - Tamanho da população: 30, 50, 100;
    - Taxa de cruzamento: 0.9, 0.95, 1.0;
    - Taxa de mutação: 0.01, 0.1, 0.5;
    - Épocas: 100, 1000, 10000;

    Lembre que os operadores genéticos também podem apresentar parâmetros e eles devem ser configurados.
6. Escolha valores iniciais S para os parâmetros dentro dos conjuntos escolhidos, por exemplo, 30, 0.9, 0.01 e 100 para o tamanho da população, taxa de cruzamento, taxa de mutação e épocas respectivamente.
7. Execute o algoritmo 30 vezes para os valores S escolhidos e registre a média da distância do percurso considerando as 30 execuções.
8. Fixe valores dos demais parâmetros e efetue 30 execuções para cada valor diferente do tamanho da população. Ao final, selecione o tamanho da população que apresenta a menor média das distâncias e atualize os valores de S.
9. Repita o passo 8 para cada um dos parâmetros de modo sequencial, isto é, o primeiro a ser variado foi o tamanho da população, depois será a taxa de cruzamento, depois a taxa de mutação e por fim a quantidade de épocas.
10. Repita o passo anterior mais uma vez considerando os valores atualizados de cada parâmetro.
11. Apresente os resultados no formato de uma tabela associando os valores dos parâmetros à média da distância e ao percurso selecionado.
12. Apresente o melhor percurso encontrado e sua distância, isto é, o melhor percurso apresentado ao final do passo 10.
13. Essa apresentação deve ser feita no arquivo readme.txt e ele deve ser submetido com o código-fonte da aplicação.