Relatório T1 Linguagens formais e compiladores.

Aluno: Gustavo Borges França.
Matrícula: 15100741

Linguagem e ferramentas utilizadas

Para a implementação do T1 foi utilizada a linguagem de programação Python e para auxiliar no desenvolvimento da interface gráfica foi utilizada a biblioteca PyQt5

Edição/Leitura/Gravação de GR e ER

A edição de GRs e ERs é feita, como foi requisitado, de forma textual, para a criação de uma nova gramática regular, deve-se selecionar no menu novo -> gramática regular. Então o usuário nomeia a gramática (um nome não vazio, e não deve existir) e entra com produções válidas para a gramática, se as produções não forem validadas pelo programa(houver algum erro) a gramática não será salva, mostrando uma mensagem de erro, se o nome existir ou estiver em branco a gramática também não será salva, se tudo ocorrer bem, a gramática será salva, e uma cópia das produções sera mostrada no campo abaixo de onde se define as produções. Tudo isso vale também para a criação de expressões, só que ao invés de validar produções, nas expressões regulares, a expressão que é validada, a "mecânica" do nome continua sendo valida aqui também. Gramáticas, expressões e autômatos são todos gravados em memória, ou seja, só existirão enquanto o programa rodar, uma vez fechado todas as gramáticas criadas somem ao abrir o programa novamente.

É possível atualizar a definição das GRs ou ERs pelo menu Editar → gramática regular ou expressão regular, e então editar utilizando os campos, tem a mesma dinâmica da criação.

Para ver as ERs ou GRs salvas basta ir no menu Listar → Gramática regular ou expressão regular, e todas as entradas em memória serão listadas, basta escolher uma para ver.

Conversão de ER em AF usando De Simone.

Não implementado.

Transformação de GR em AF e vice-versa

Para converter basta acessar o menu converter e escolher uma opção, de GR para AF ou AF para GR. Os algoritmos utilizados são aqueles que foram vistos em aula, basta escolher um GR(AF) na lista para converter em AF(GR)

De GR → AF

Os não terminais viram os estados

Os terminais são o alfabeto

Um estado $ é criado e é final, ele não tem transições, ou melhor, vai para erro por qualquer.

As produções do tipo S→aB viram delta(S, a) = B

produções do tipo S→a viram delta(S,a) = $

se S→& então S é um estado final (considerando todas as condições para & aparecer em uma GR).

De AF→GR
Os estados viram os não terminais
O alfabeto vira os terminais
as transições delta(S,a) = A viram produções S→aA e se A for de aceitação S->aA|a
Se o estado inicial for de aceitação S→&

Determinização e Minimização de AF

A determinização e minimização de AF podem ser realizadas pelo menu de visualização de AF Listar → autômatos

Os autômatos que não são determinísticos mostrarão um botão para determinizar e os que são determinísticos mostram um botão para minimizar.

O algoritmo de determinização é o mesmo que foi visto em aula.
Copia o estado inicial e “funde” as transições por símbolo
Adiciona essas “fusões” como estados novos (determinísticos)
a partir desses estados são definidas suas transições novas, que são as “fusões” das transições dos estados que foram unidos
cada estado novo que surge repete.

No fim será obtido um Autômato finito determinístico.

Para minimizar o algoritmo utilizado foi o algoritmo de Brzozowski.

Esse Algoritmo diz que dado um AFD de uma Linguagem, ao inverter esse AFD, determinizar a inversão e remover estados inalcançáveis, obtém-se um AFD mínimo para o inverso da linguagem original, ao repetir todas as operações mais uma vez obtém-se o AFD mínimo para a linguagem original.

Dado AFD M então
um M’ minímo seria obtido com alc(det(rev(alc(det(rev(M))))))

O algoritmo de inversão é
cria um novo estado inicial.
se o estado inicial de M é final o estado inicial de M' (novo estado criado)
também é.
para cada estado então as transições são invertidas
se A vai para B por a em M, B vai para A por a em M'
os finais de M não são mais finais em M'
o inicial de M agora é final em M'
o estado inicial novo de M' clona as transições dos que eram finais em M

O algoritmo para remover estados inalcançáveis é
remoção de estados inalcançáveis.
cria uma lista de estados alcançáveis.
partindo de q0 os estados que são alcançáveis são adicionados a esta lista
remove os que não estão nessa lista.

AF que representa intersecção, diferença e reverso de LR.
LR dada por GR, AF ou ER

Como De Simone não foi implementado, a definição de LR por ER também não.

Para obter o AF para alguma dessas operações, basta ir em operações → linguagem regular e escolher a operação e como será definida a LR.

O algoritmo do reverso foi explicado anteriormente.

Para realizar as operações de intersecção e diferença devem ser implementados algoritmos para realizar união e complemento de AF.

União
Cria um novo estado inicial @
esse estado inicial clona as transições dos estados iniciais de M1 e M2 a serem unidos.
Os iniciais de M1 e M2 deixam de ser iniciais, o autômato resultante é AFN para M1 U M2

Complemento
Completa o autômato
Os estados de aceitação viram de não aceitação.
Os estados de não aceitação viram de aceitação.
Resultado é M' que é not(M)

Intersecção é definida como not(not(M1) U not(M2))
então basta determinizar a união dos complementos de M1 e M2
depois complementar essa União que será obtido a intersecção.

A diferença é definida como M1 ∩ not(M2)
então basta realizar a intersecção de M1 com o complemento de M2.

Se o método para definir a LR for gramática regular a gramática é primeiramente transformada em um AF pela conversão definida lá em cima. Se for por AF, é feito direto.

Obtenção de GR da união, concatenação e fecho de GR

Para obter a união de GR, ou qualquer outra operação basta ir ao menu de operações → gramática regular → qualquer operação desejada.

Para a união de GR
dado duas GRs com conjuntos de não terminais diferentes
cria um novo não terminal @ que será o símbolo inicial das produções.
Se épsilon tá presente em alguma das GRs, @→ & e & é removido das GRs que ele está presente
As produções inicias de cada GR são copiadas para @
a gramática resultante é a união

Para a concatenação de GR
dadas duas GRs com conjuntos de não terminais diferentes
Todas as produções da GR 1 no formato S→a agora se tornam S1 → aS2
GR resultante é a concatenação.

Para estrela
Dada uma GR
toda produção do tipo S→a deve ser expandida para S → a|aS
um novo estado inicial deve ser criado S1 → & | produções do simbolo inicial anterior
GR resultante é o fechamento da GR anterior.

Reconhecimento de sentenças aceitas por um AF e enumeração de sentenças de tamanho n aceitas por ele

Para verificar se um autômato reconhece uma sentença basta ir no menu operações → reconhecer entrada. Após basta escolher um autômato e passar uma entrada. Na tela aparecerá o resultado se foi aceita ou rejeitada a entrada.

Dado um AFD completo
Estado atual é o estado inicial.
Para cada c da entrada dada
    o estado atual é agora o estado para qual o estado atual anterior ia pelo simbolo c
ao fim da entrada se o estado atual for de aceitação
    retorna aceitou
senão retorna rejeitou

Para enumerar sentenças de tamanho n aceitas pelo AF basta ir no menu de operações e ir em enumerar sentenças. Escolhido um autômato basta dar um tamanho n e serão geradas todas as combinações de tamanho n possíveis das letras do alfabeto do AF e essas sentenças geradas são testadas no algoritmo de cima, se aceita ela entra para uma lista e quando terminado o processo, a lista das sentenças é apresentada ao usuário.