TEMA Tend. Mat. Apl. Comput., **4**, No. 1 (2003), 109-118.

O Modelo de Máquina Geométrica Intervalar[1]

R.H.S. REISER, A.C.R. COSTA, G.P. DIMURO[2], ESIN, NAPI, UCPel, Cx.P. 420, 96010-140 Pelotas, RS, Brasil.

Resumo. Mostra-se neste trabalho que a linguagem e a correspondente semântica associada às interpretações obtidas na estrutura ordenada da Máquina Geométrica Intervalar, fundamentada nos espaços coerentes, são ferramentas importantes para construção, desenvolvimento e análise semântica de algoritmos da matemática intervalar, envolvendo paralelismo e não-determinismo, definidos por estruturas matriciais e operando de forma sincronizada.

1. Introdução

Fundamentos da teoria clássica da computação são diretamente aplicados aos problemas da matemática computacional, da análise numérica e da computação científica. A busca do controle automático, rigoroso e completo dos erros, com limites confiáveis, justificam o uso de técnicas intervalares no desenvolvimento e aplicação de sofwares para Computação Científica [5]. Portanto, é de grande interesse o estudo e análise de sistemas paralelos e distribuídos aplicados às computações intervalares. Inspirada na teoria dos sistemas dinâmicos, a idéia intuitiva do modelo Máquina Geométrica Intervalar (MGI) é a modelagem da memória (representando intervalos) e dos processos computacionais (representando operações sobre intervalos) por pontos do espaço euclidiano tridimensional. A noção de processo é a de uma transformação entre estados de computação, que pode ser aplicada à construções não-determinísticas, incluindo processos concorrentes.

2. A estrutura ordenada do modelo MGI

Nesta seção, os principais aspectos relacionados com a estrutura ordenada do modelo MGI são apresentados, incluindo o espaço coerente dos processos computacionais $\mathbb{D}_\infty$, o espaço coerente dos estados computacionais $\mathbb{S}$ e o espaço coerente dos testes computacionais $\mathbb{B}$, definidos a partir do espaço coerente plano dos nomes (ou posições) $\mathbb{R}^3$ e do espaço coerente dos valores $\mathbb{IQ}$. No conjunto dos intervalos de reais IR definem-se os dados de entrada e de saída no modelo MGI.

[1]Trabalho parcialmente financiado por CTINFO-CNPq e FAPERGS.
[2]{reiser,rocha,liz}@atlas.ucpel.tche.br PPGC, UFRGS, Cx.P. 15064, 91501-970 Porto Alegre, RS, Brasil.

Os conceitos e definições apresentados logo a seguir referem-se à Teoria dos Domínios [4], mais especificamente aos Espaços Coerentes, domínios algébricos caracterizados pela compatibilidade binária, introduzidos por Girard [3] na formalização de uma semântica denotacional para a Lógica Linear.

2.1. Espaços coerentes e funções lineares

Uma teia $\mathbf{W} = (W, \approx_W)$ é um par consistindo de um conjunto W e uma relação reflexiva e simétrica $\approx_W$, chamada relação de coerência. Um subconjunto desta teia satisfazendo a compatibilidade binária é denominado um subconjunto coerente. A coleção de subconjuntos coerentes da teia $\mathbf{W}$, parcialmente ordenada pela relação de inclusão, é denominada *espaço coerente*, e indicada por $\mathbb{W} \equiv (Coh(\mathbf{W}), \subseteq)$ [12]. Em particular, $|\mathbb{W}| = \{w \mid \{w\} \in \mathbb{W}\} = W$.

Funções lineares são funções contínuas no sentido proposto por Scott [11] que também satisfazem às propriedades de estabilidade e linearidade, assegurando a existência e unicidade da menor aproximação na imagem. Considerando os espaços coerentes $\mathbb{A}$ e $\mathbb{B}$, uma função linear $f : \mathbb{A} \to \mathbb{B}$ é identificada por seu traço linear, um subconjunto de $A \times B$ dado por $ltr(f) = \{(\alpha, \beta) \mid \beta \in f(\alpha)\}$.

Seja $\mathbf{A} \multimap \mathbf{B} = (A \times B, \approx_{\multimap})$ a teia de traços lineares com a relação de coerência definida por $(\alpha, \beta) \approx_{\multimap} (\alpha', \beta') \leftrightarrow ((\alpha \approx_{\mathbf{A}} \alpha' \to \beta \approx_{\mathbf{B}} \beta')$ e $(\beta = \beta' \to \alpha = \alpha'))$. A coleção de conjuntos coerentes da teia $\mathbf{A} \multimap \mathbf{B}$, ordenada pela inclusão, define o domínio $\mathbb{A} \multimap \mathbb{B}$ dos traços lineares das funções definidas de $\mathbb{A}$ para $\mathbb{B}$.

Compatível com a idéia de que a memória não é necessariamente limitada, o modelo MGI interpreta estados e processos possivelmente infinitos, representados por coleções de traços lineares de funções lineares. Neste caso, cada traço linear modela o comportamento deste processo num instante de tempo computacional e constitui-se numa aproximação parcial da função global computada pelo processo. O tempo associado a uma computação é medido pelo número de processos elementares executados (unidade computacional de tempo - *uct*). O estado final é formalmente definido como o limite das aproximações parciais que constituem as diversas etapas de computação, caracterizando a completação da estrutura ordenada do modelo.

2.2. O espaço coerente dos estados computacionais

A noção de estado de computação no modelo MGI introduz um conjunto de variáveis distribuídas no espaço geométrico, às quais são atribuídos valores (no conjunto dos números reais extendidos R) e rótulos (indicando posições no espaço euclidiano tridimensional R^3). Assim, considera-se a construção dos reais computáveis utilizando o Espaço Coerente Bi-Estruturado de Intervalos Racionais, $\mathbb{IQ}$ [2]. A representação global e construtiva para os números reais e intervalos de reais em $\mathbb{IQ}$ é realizada em dois níveis. Compreende a construção interna dos objetos da estrutura de informação do domínio $\mathbb{IQ}$ e sua estrutura externa de aplicação, representando as operações algébricas, a ordem de posição e as funções elementares [6].

Seja $\mathbb{R}^3$ o domínio plano interpretando os pontos do espaço euclidiano tridimensional. Numa abordagem não-determinística, cada estado computacional é inter-

pretado por uma coleção de traços lineares. Formalmente,

Definição 1. *Seja* $S \equiv \mathbb{R}^3 \multimap \mathbb{IQ}$ *o espaço coerente dos traços lineares de funções lineares definidas do espaço coerente dos nomes* $\mathbb{R}^3$ *para o espaço coerente dos valores* $\mathbb{IQ}$*, no modelo MGI.* $\mathbf{S} = (Coh(S), \approx_S)$ *indica a teia cujos elementos são subconjuntos coerentes de* $\mathbb{R}^3 \multimap \mathbb{IQ}$ *relacionadas pela relação trivial de coerência, indicada por* $\approx_S$*. A coleção dos subconjuntos coerentes de* $\mathbf{S}$*, parcialmente ordenada pela inclusão, define o espaço coerente dos estados,* $\mathbb{S} \equiv (Coh(\mathbf{S}), \subseteq)$.

A Definição 1 inclui a abordagem determinística: $\mathbf{s} \in \mathbb{S}$ é um estado determinístico se, e somente se, $\mathbf{s} = \{s\}$ e $s \in |\mathbb{S}| = S$.

2.3. O espaço coerente dos testes computacionais

No modelo MGI, o espaço coerente dos testes fundamenta-se na lógica binária. Para tal, considera-se primeiramente a teia discreta $\mathbf{Bool} \equiv (Bool, =)$, dada pelo conjunto $Bool = \{V, F\}$ de valores booleanos e a relação de igualdade. A coleção de todos os subconjuntos coerentes $(Coh(\mathbf{Bool} = \{\emptyset, \{V\}, \{F\}\})$, ordenada pela inclusão, define o espaço coerente dos valores booleanos $\mathbb{V}_{Bool}$.

No espaço coerente $\mathbb{B} \equiv \mathbb{S} \multimap \mathbb{V}_{Bool}$ são modelados os testes booleanos. A abordagem não-determinística conduz os estudos a um tratamento não tradicional dos testes. Para cada teste determinístico b são considerados duas formas distintas de testes $\mathbf{b}$ sobre estados não-determinísticos $\mathbf{s}$:

1. a forma universal $(\mathbf{b}_\forall(\mathbf{s}) \equiv \forall s \in \mathbf{s} \,.\, b(s))$ e
2. a forma existencial $(\mathbf{b}_\exists(\mathbf{s}) \equiv \forall s \in \mathbf{s} \,.\, b(s))$.

Ambos coincidem com o teste b quando $\mathbf{s}$ é um estado determinístico $(\mathbf{s} = \{s\})$.

2.4. O espaço coerente das transições de estados

A partir do conjunto dos traços lineares das funções lineares definidas sobre o espaço coerente $\mathbb{S}$ define-se o espaço coerente $\mathbb{S} \multimap \mathbb{S}$, interpretando as transições de estados. Cada conjunto coerente em $\mathbb{D}_\infty$, modelando um processo, está associado a uma função linear, cujo traço linear é um objeto de $\mathbb{S} \multimap \mathbb{S}$. Assim, pela análise semântica obtida pelos traços lineares das funções em $\mathbb{S} \multimap \mathbb{S}$ se obtém uma análise do comportamento dos processos, compatível com a construção indutiva e ordenada dos objetos no espaço coerente $\mathbb{D}_\infty$. De acordo com a noção de processo como transformação de estados, no espaço coerente $\mathbb{S} \multimap \mathbb{S}$ o significado dos processos interpretados por objetos em $\mathbb{D}_\infty$ pode ser descrito em termos de seqüências de etapas de computações formalizando uma semântica baseada na transição de estados. Em [7] mostra-se que a semântica denotacional associada à construção dos processos no modelo MG é obtida pela defição da função de avaliação.

Proposição 1. *Seja* $\mathbb{A} \equiv S \multimap \mathbb{IQ}$ *o espaço coerente das ações computacionais. Se* $\mathrm{d}, \mathrm{pr}^{(k)} \in \mathbb{A}$*, com* $\mathrm{pr}^{(k)}(s) = s(k)$*, então é linear a função* $d^{(k)} \in S \multimap S$ *dada por*

$$d^{(k)}(s)(i) = \begin{cases} \mathrm{pr}^{(i)}(s) = s(i), & \textit{se } i \neq k, \\ \mathrm{d}(s), & \textit{caso contrário.} \end{cases}$$

Prova. Seja $s', z' \in S \equiv \mathbb{R} \multimap \mathbb{IQ}$, $i, k \in \mathbb{IQ}$ e $s' = \mathrm{d}^{(k)}(s), z' = \mathrm{d}^{(k)}(z)$. Mostra-se que, para todo par $(s, s'), (z, z')$ no grafo da função $\mathrm{d}^{(k)}$ as seguintes propriedades são satisfeitas. (1) Se $s \approx_{\multimap} z$ então $s' \approx_{\multimap} z'$: Suponha $s \approx_{\multimap} z$. Se $i \neq k$ e $\mathrm{d} \in [\mathbb{R} \multimap \mathbb{IQ}] \multimap \mathbb{IQ}$ então $\mathrm{d}^{(k)}(s)(i) = s(i) \approx_{\mathbf{IQ}} z(i) = \mathrm{d}^{(k)}(z)(i)$. Caso contrário, se $i = k$ então $\mathrm{d}^{(k)}(s)(k) = \mathrm{d}(s) \approx_{\multimap} \mathrm{d}(z) = \mathrm{d}^{(k)}(z)(k)$. Logo, $s' \approx_{\multimap} z'$. (2) Se $s' = z'$ então $s = z$: Suponha $s' = z'$. Quando $i \neq k$ e $\mathrm{d} \in \mathbb{A}$ é uma função linear, $s(i) = \mathrm{d}^{(k)}(s)(i) = s'(i) = z'(i) = \mathrm{d}^{(k)}(z)(i) = z(i)$ então $s = z$. Por outro lado, se $i = k$, $\mathrm{d}^{(k)}(s)(i) = \mathrm{d}(s)(k) = \mathrm{d}(z)(k) = \mathrm{d}^{(k)}(z)(i)$ e $s = z$, de acordo com a linearidade da função d. Portanto, em qualquer um dos casos, $s = z$ e $\mathrm{d}^{(k)}$ é linear.

Proposição 2. *Para cada função* $\mathbf{p} \in \mathbb{S} \multimap \mathbb{S}$, $p \in S \multimap S$ *tem-se* $\mathbf{p}(\mathbf{s}) = \{p(s) \mid s \in \mathbf{s} \subseteq |\mathbb{S}|\}$.

Numa abordagem determinística, pela Proposição 2, $\mathbf{p}(\{s\}) = \{p(s)\}$ se $s \in |\mathbb{S}|$. A seguir, apresenta-se a função que indica o conjunto de posições de memória afetados após a execução de um processo. Os próximos parágrafos definem as funções de transição que modelam processos, testes e construtores no modelo MGI.

Definição 2. *Sejam* $\mathbf{p} \in \mathbb{S} \multimap \mathbb{S}$, $p \in S \multimap S$ *satisfazendo a Proposição 2. A função-posição* $\Upsilon_{\mathbb{S}} : [\mathbb{S} \multimap \mathbb{S}] \to \wp(\mathrm{R}^3)$ *é dada por* $\Upsilon_{\mathbb{S}}(\mathbf{p}) = \bigcup_{s \in \mathbf{s}} \{i \mid p(s)(i) \neq s(i)\}$.

Definição 3. *Sejam* $\mathrm{d}^{(k)} \in S \to S$ *conforme Proposição 1, a função identidade* $Id_{\mathbb{S} \multimap \mathbb{S}}$, $\mathbf{s} = \{s(i)\}_{i \in \mathbb{R}^3} \in \mathbf{S}$ *tal que* $s \in |\mathbb{S}|$. *No modelo MGI, define-se:*

1. *processo elementar* $\mathbf{d^k} \in \mathbb{S} \multimap \mathbb{S}$ *por* $\mathbf{d^k}(\mathbf{s}) = \{\mathrm{d}^{(k)}(s) \mid s \in \mathbf{s}\}$;
2. *teste determinístico* $b : S \to \mathbb{B}$, *por* $b(s) = \begin{cases} \{V\}, & \text{se } b(s) \text{ é satisfeito,} \\ \{F\}, & \text{se } b(s) \text{ não é satisfeito,} \\ \emptyset, & \text{caso contrário;} \end{cases}$
3. *teste universal* $\mathbf{b}_\forall : \mathbb{S} \to \mathbb{B}$, *é definido por* $\mathbf{b}_\forall(\mathbf{s}) = \begin{cases} \{V\}, & \text{se } \forall s \in \mathbf{s}, b(s) = \{V\}, \\ \{F\}, & \text{caso contrário;} \end{cases}$
4. *teste existencial* $\mathbf{b}_\exists : \mathbb{S} \to \mathbb{B}$, *por* $\mathbf{b}_\exists(\mathbf{s}) = \begin{cases} \{V\}, & \text{se } \exists s \in \mathbf{s}, b(s) = \{V\}, \\ \{F\}, & \text{caso contrário;} \end{cases}$
5. *composição seqüencial* $\circ : [\mathbb{S} \multimap \mathbb{S}]^2 \to [\mathbb{S} \multimap \mathbb{S}]$, $(\mathbf{p} \circ \mathbf{q})(\mathbf{s}) = \bigcup_{s \in \mathbf{s}} \{\mathbf{q} \circ \mathbf{p})(\{s\})\}$,
6. *escolha condicional* $\circ_+ : [\mathbb{S} \multimap \mathbb{B}] \times [\mathbb{S} \multimap \mathbb{S}]^2 \to [\mathbb{S} \multimap \mathbb{S}]$ *dada por*
$$\circ_+(\mathbf{b}_\alpha, \mathbf{p}, \mathbf{q})(\mathbf{s}) = \begin{cases} \mathbf{p}(\mathbf{s}), & \text{se } \mathbf{b}_\alpha(\mathbf{s}) = \{V\}, \\ \mathbf{q}(\mathbf{s}), & \text{se } \mathbf{b}_\alpha(\mathbf{s}) = \{F\}, \\ Id(\mathbf{s}), & \text{caso contrário;} \end{cases} \quad \text{sempre que } \alpha \in \{\forall, \exists\}.$$
7. *composição paralela* $\circ_\| : [\mathbb{S} \multimap \mathbb{S}]^2 \to [\mathbb{S} \multimap \mathbb{S}]$, *dada pelos casos:*
   1. *se* $\Upsilon_{\mathbb{S}}(\mathbf{p}) \bigcap \Upsilon_{\mathbb{S}}(\mathbf{q}) = \emptyset$ *então* $(\mathbf{p} \circ_\| \mathbf{q})(\mathbf{s}) = \bigcup_{s \in \mathbf{s}} (\mathbf{p} \circ_\| \mathbf{q})\{s\}$, *sempre que*
   $$(\mathbf{p} \circ_\| \mathbf{q})\{s\}(i) = \begin{cases} \mathbf{p}\{s\}(i), & \text{se } i \in \Upsilon_{\mathbb{S}}(\mathbf{p}), \\ \mathbf{q}\{s\}(i), & \text{se } i \in \Upsilon_{\mathbb{S}}(\mathbf{q}), \\ \{s(i)\} & \text{caso contrário;} \end{cases}$$
   2. *senão* $(\mathbf{p} \circ_\| \mathbf{q}) = Id_{\mathbb{S} \multimap \mathbb{S}}$.

8. *A escolha não-determinística* $\circ_| : [\mathbf{S} \multimap \mathbf{S}]^2 \to [\mathbf{S} \multimap \mathbf{S}]$ *dada por casos:*
   1. $(\mathbf{p} \circ_| \mathbf{q})(\mathbf{s}) = \mathbf{p}(\mathbf{s}) \bigcup \mathbf{q}(\mathbf{s})$, *se* $\Upsilon_{\mathbf{S}}(\mathbf{p}) \bigcap \Upsilon_{\mathbf{S}}(\mathbf{q}) \neq \emptyset$*;*
   2. $\mathbf{p} \circ_| \mathbf{q} = Id_{\mathbb{S} \multimap \mathbb{S}}$, *caso contrário.*

2.5. O espaço coerente dos processos computacionais

Esta seção resume a construção indutiva do espaço coerente dos processos $\mathbb{D}_\infty$ apresentada em [8]. Seguindo a metodologia sugerida por Scott [11], cada nível da construção caracteriza um subespaço coerente $\mathbb{D}_n$, definido a partir do relacionamento baseado na coerência binária entre unidades atômicas de informações de sua teia geradora – grafo reflexivo e não-dirigido, cujos tokens interpretam processos. De acordo com os conceitos apresentados na Seção 2.4., cada subconjunto de tokens coerentes interpretando um processo em $\mathbb{D}_n$, e posteriormente imerso em $\mathbb{D}_\infty$, está associado a uma função linear em $\mathbb{S} \multimap \mathbb{S}$ modelando seu comportamento.

Tomando-se como base o espaço corente $\mathbb{D}_0$ dos processos elementares, define-se o espaço coerente $\mathbb{D}_1$, no qual são interpretados todos os processos executados em no máximo 2^1 *uct*. A metodologia de construção do primeiro nível $\mathbb{D}_0 - \mathbb{D}_1$ é repetida na construção dos demais níveis. Cada nível da construção, identificado por um subespaço $\mathbb{D}_n$, que reconstrói todos os objetos do nível anterior, preservando suas propriedades e relações, além de construir os novos objetos. Compatível com a abordagem algébrica, o relacionamento entre os níveis é expresso por funções lineares denominadas imersões e projeções, interpretanto os construtores de processos e seus destrutores, respectivamente. Pelo procedimento de completação, assegura-se a existência do menor ponto fixo para equações recursivas definidas pela composição (possivelmente infinita) destes morfismos. Além disso, as interpretações para processos infinitos, construídos por prefixação, apresentadas em $\mathbb{D}_\infty$ comprovam que este modelo é compatível com a diversidade dos construtores. Em $\mathbb{D}_n - \mathbb{D}_{n+1}$ são interpretados os processos que executam no máximo 2^{n+1} operações externas (composições seqüenciais e condicionais) ou um número arbitrário de operações internas (composições não-determinísticas ou síncronas) interpretadas como objetos construídos em $\mathbb{P}_n$. Neste caso, $\mathbb{P}_n$ é obtido pela aplicação do operador categórico soma direta [12] ($\mathbb{P}_n = \mathbb{D}_n \coprod \bar{\mathbb{D}}_n \coprod \bar{\mathbb{D}}_n^\perp$), que justapõe os subespaços coerentes $\mathbb{D}_n$, $\bar{\mathbb{D}}_n$ e $\bar{\mathbb{D}}_n^\perp$ interpretando as composições de processos finitos ($2^n uct$), as composições paralelas e as escolhas não-determinísticas, respectivamente. Salienta-se que, a relação de coerência que constrói a teia sobre a qual se define o espaço coerente $\bar{\mathbb{D}}_n$ é obtida a partir da definição recursiva da função-posição $\Upsilon_{\mathbb{D}_n}$, ao explicitar o conjunto de posições de memória alteradas por cada processo, identificando a composição paralela. No espaço dual, $\bar{\mathbb{D}}_n^\perp$ tem-se a interpretação para a relação de conflito de acesso a posições na memória, característica das escolhas não-determinísticas.

Por fim, a aplicação do operador produto direto constrói as interpretações para a composição seqüencial em $\mathbb{P}_n \prod \mathbb{P}_n$ e, considerando-se o espaço coerente $\mathbb{B}$, composição condicional interpretada no subespaço $\mathbb{P}_n \prod_{\mathbb{B}} \mathbb{P}_n$. Portanto, cada subespaço é gerado pelas sucessivas iterações da equação:

$$\mathbb{D}_{n+1} = \mathbb{P}_n \coprod (\mathbb{P}_n \prod \mathbb{P}_n) \coprod (\mathbb{P}_n \prod_{\mathbb{B}} \mathbb{P}_n), \text{ se } \mathbb{P}_n = \mathbb{D}_n \coprod \bar{\mathbb{D}}_n \coprod \bar{\mathbb{D}}_n^\perp.$$

O espaço coerente $\mathbb{D}_\infty$ constitui-se no menor ponto fixo para a equação anterior, que define o espaço coerente dos processos finitos [1].

2.6. O Modelo MGI

Neste trabalho, os números reais que indicam o centro e o raio de cada intervalo $a \in \mathrm{IR}$ são os valores associados a pontos em R^3, respectivamente indicados por $a_c = s(l,i,0)$, $a_r = s(l,i,1) \geq 0$. Por conseqüência, cada intervalo real $a = [a_c, a_r] \in \mathrm{IR}$ está associado a uma posição de memória bi-dimensional indicada por $s(l,i) = a$, $(l,i) \in \mathrm{R}^2$. Analogamente, uma matriz intervalar será modelada por um par de matrizes reais, cuja segunda componente é sempre uma matriz não-negativa (matriz dos raios dos intervalos). A próxima definição é construída de acordo com a definição de máquina proposta em [10].

Definição 4. *Considere os espaços coerentes* $\mathbb{S}$, $\mathbb{B}\,\mathbb{S} \multimap \mathbb{S}$ *e* $\mathbb{D}_\infty$. *O modelo MGI é uma função* $\mathcal{M}$ *cujo domínio é o conjunto* $\{\mathbf{in}\} \bigcup \mathbb{D}_\infty \bigcup \mathbb{B} \bigcup \{\mathbf{out}\}$, *tal que*

1. $\mathcal{M}_{\{\mathbf{in}\}} : \{\mathbf{in}\} \to [\mathrm{IR} \to \mathbb{S}]$. *Em particular,*

$$\mathcal{M}_{\{\mathbf{in}\}}(\mathbf{in})([a_c, a_r]) = \{\{v_i\}_{i \in \mathbb{R}^3}\},\ v_i = \begin{cases} x_{a_c} \in \mathbb{IQ}, & \text{se } i = \{(0,0,0)\}, \\ x_{a_r} \in \mathbb{IQ}, & \text{se } i = \{(0,0,1)\}, \\ x_0 \in \mathbb{IQ}, & \text{caso contrário;} \end{cases}$$

2. $\mathcal{M}_{\mathbb{D}_\infty} : \mathbb{D}_\infty \to [\mathbb{S} \multimap \mathbb{S}]$, $\mathcal{M}_{\mathbb{D}_\infty}(x)(\mathbf{s}) = x(\mathbf{s})$, *conforme Definição 3;*

3. $\mathcal{M}_{\mathbb{B}}(\mathbf{b}_\alpha) : \mathbb{B} \to [\mathbb{S} \multimap \mathbb{V}_{bool}]$, *com* $\alpha \in \{\forall, \exists\}$ *e* $\mathcal{M}_{\mathbb{B}}(\mathbf{b}_\alpha)(\mathbf{s}) = \mathbf{b}_\alpha(\mathbf{s})$ *conforme Definição 3;*

4. $\mathcal{M}_{\{\mathbf{out}\}} : \{\mathbf{out}\} \to [\mathbb{S} \to \wp(\mathrm{IR})]$, $(M)_{\mathbf{out}}(\mathbf{s}) = \{[s(i,j,0), s(i,j,1)] \mid s \in \mathbf{s}\}$.

Considerado um estado $\mathbf{s}$, a execução de um processo no modelo MGI é dada por: $\mathcal{M}(x)(\mathbf{s}) = (M)_{\{\mathbf{in}\}} \circ (M)_{\{\mathbb{D}_\infty\}} \circ (M)_{\{\mathbf{out}\}}(\mathbf{s})$.

3. A linguagem para o modelo MGI

Neste seção, apresenta-se um resumo da linguagem $\mathcal{L}(\mathbb{D}_\infty)$ induzida pelas interpretações obtidas no domínio $\mathbb{D}_\infty$ [9]. Para tal, sejam $\mathcal{S}$ o conjunto de expressões de acesso aos estados e $\mathcal{K}$ o conjunto de símbolos constantes dado pela união $\mathcal{K} = I_T \bigcup \{\mathbf{skip}\} \bigcup \{C\}$, onde I_T e I_C indicam os conjuntos de símbolos representando testes, incluindo identificador para processo identidade(**skip**) e valores constantes do modelo MGI, respectivamente. I_{PEl} indica o conjunto se símbolos de identificadores de processos elementares, onde cada processo elementar é dado por $s[i] :=$*expressão* sende $s[i]$ é uma expressão de acesso ao estado no modelo MGI.

Se I_P indica o conjunto de identificadores de processos, $F_{Op} = \{Id, \|\,, |,\, \cdot\,, +\}$ é o conjunto de indicadores dos construtores de expressões de processos, onde:
(i) $Id \in I_P$ indica o construtor identidade;
(ii) $\cdot, \|, |\colon I_P \times I_P \to I_P$ são símbolos representando as composições seqüencial e

paralela e escolha não-determinística apresentadas na Seção 2.4., respectivamente;
(iii) $+ : I_P \times I_P \times I_T \to I_P$ é um símbolo de aridade 3, representando a escolha determinística (ou ainda $+_b : I_P \times I_P \to I_P$, $\forall b \in I_T$). O conjunto de expressões da linguagem $\mathcal{L}(\mathbb{D}_\infty)$ é definido por:
(i) expressões de estado símbolos constantes são expressões em $\mathcal{L}(\mathbb{D}_\infty)$;
(ii) se $* \in \{\|, |, \cdot, +_b\}$ e $t_0, t_1, \ldots, t_n, t_{n+1}, \ldots \in \mathcal{L}(\mathbb{D}_\infty)$, então

$$*_{i=n+1}^{0}(t_i) = t_{n+1} * t_n * \ldots * t_0 \quad \text{e} \quad *_{i=0}^{n+1}(t_i) = t_0 * \ldots * t_n * t_{n+1}$$

são expressões (finitas) em $\mathcal{L}(\mathbb{D}_\infty)$;
(iii) se $t_0, t_1, \ldots, t_n, t_{n+1}, \ldots \in \mathcal{L}(\mathbb{D}_\infty)$ então $*_{n=0}^{\infty} t_n = t_0 * t_1 * \ldots * t_{n+1} * \ldots$ é uma expressão infinita em $\mathcal{L}(\mathbb{D}_\infty)$.

Diz-se que cada expressão em $\mathcal{L}(\mathbb{D}_\infty)$ é uma representação de um processo e seu correspondente conjunto coerente em $\mathbb{D}_\infty$. A interpretação das expressões na MGI é dada pela função $eval : \mathcal{L}(\mathbb{D}_\infty) \to (\mathbb{S} \multimap \mathbb{S})$ tal que $eval(x)(\mathbf{s}) = \mathcal{M}(x)(\mathbf{s})$.

4. Programando no Modelo MGI

Operações aritméticas envolvendo intervalos e vetores (matrizes) de intervalos serão apresentadas nesta seção. Neste caso, cada intervalo é dado por um par de números reais indicando seu centro e seu raio, tornando mais intutitiva a mensuração do erro nas computações associadas aos algoritmos que utilizam a aritmética intervalar [3].

Proposição 3. *Sejam* $a_c, a_r, b_c, b_r \in \mathrm{R}_\infty$, $a = [a_c, a_r], b = [b_c, b_r] \in \Im$ *e o conjunto* $M = \{m_1, m_2, m_3, m_4\}$ *onde*

$$m_1 = a_c \cdot b_c + a_r \cdot b_c + a_c \cdot b_r + a_c \cdot b_c, \quad m_2 = a_c \cdot b_c - a_r \cdot b_c + a_c \cdot b_r + a_c \cdot b_c,$$
$$m_3 = a_c \cdot b_c + a_r \cdot b_c - a_c \cdot b_r + a_c \cdot b_c, \quad m_4 = a_c \cdot b_c - a_r \cdot b_c - a_c \cdot b_r + a_c \cdot b_c.$$

As operações aritméticas $+, -, \cdot, / : \mathrm{IR}^2 \to \mathrm{IR}$ *são definidas pelas expressões:*

$$a + b = [a_c + b_c, a_r + b_r], \qquad a - b = [a_c - b_c, a_r + b_r],$$
$$a \cdot b = [min(M), max(M)], \quad a/b = a \cdot b^{-1}, \quad se\ 0 \notin b\ e\ b^{-1} = (b_c^2 - c_r^2)^{-1}[b_c, b_r].$$

Prova. É imediata, basta considerar para o intervalo $a = [\underline{a}, \overline{a}]$, as igualdades $a_c = \frac{\underline{a}+\overline{a}}{2}$ e $a_r \frac{\underline{a}-\overline{a}}{2}$ que definem seu centro e raio.

Sempre que o conjunto de rótulos I, indicando posições no espaço geométrico R^3 que definem processos e estados no modelo MGI, pode ser descrito por um processo indutivo, a relação de concorrência (conflito) responsável pelo composição paralela (escolha não-determinística), pode ser explicitada por equações recursivas. Por outro lado, a natureza indutiva do modelo e sua completação asseguram também interpretação para as composições seqüencial e escolha determinística, através de operadores lineares recursivamente definidos sobre o espaço coerente $\mathbb{D}_\infty$. Portanto, o modelo contempla interpretação para recursão no sentido espacial e temporal.

[3] A notação é a mesma para os operadores em R e IR.

Suponha que o espaço geométrico é uma discretização do espaço euclidiano tridimensional onde $I = \omega^3 \subseteq \mathrm{R}^3$. Neste caso, indica-se o conteúdo de cada posição de memória por $s[l, i, u]$, $l, i, u \in \omega$. Cada intervalo é identificado pelas coordenadas $(l, i) \in \omega^2$, sendo que $s[l, i, 0], s[l, i, 1] \in \mathrm{R}$ são os valores associados ao seu centro e seu raio, respectivamente. Exemplificações de somas e subtrações com intervalos são definidos, aplicando os construtores modelados em MGI e considerando-se os seguintes processos elementares e respectivas representações.

$\mathcal{L}(\mathbb{D}_\infty)$	$\mathbb{D}_\infty$
$\bullet\texttt{sum\_c}(l,m,n,i,j,k) \equiv (s[l,i,0] := s[m,j,0] + s[n,k,0])$	$\{sum\_c^{(li0)}_{:00}\}$
$\bullet\texttt{sum\_r}(l,m,n,i,j,k) \equiv (s[l,i,1] := s[m,j,1] + s[n,k,1])$	$\{sum\_r^{(li1)}_{:00}\}$
$\bullet\texttt{sub\_r}(l,m,n,i,j,k) \equiv (s[l,i,1] := s[m,j,1] - s[n,k,1])$	$\{sub\_r^{(li1)}_{:00}\}$.

1. $\texttt{sum}(l,m,n,i,j,k) \equiv \texttt{sum\_c}(l,m,n,i,j,k) \parallel \texttt{sum\_r}(l,m,n,i,j,k)$ é a expressão do processo que executa, em 1 *uct*, a soma dos intervalos nas posições de memória (m, j) e (n, k), colocando o intervalo resultante na posição de memória (l, i). Sua representação no domínio $\mathbb{D}_\infty^{\rightarrow}$ é dada pelo conjunto $x_i = \{\overline{sum\_c^{(li0)}}_{10:00}, \overline{sum\_r^{(li1)}}_{10:00}\} = F_{(10)}(\{\overline{sum\_c^{(li0)}}_{10:00}\} \sqcap \{\overline{sum\_r^{(li1)}}_{10:00}\})$.
   Na subtração de intervalos, observações análogas são obtidas para o processo $\texttt{sub}(l,m,n,i,j,k) \equiv \texttt{sum\_c}(l,m,n,i,j,k) \parallel \texttt{sub\_r}(l,m,n,i,j,k)$.

2. Se $\texttt{sum\_row}(l,m,n,i) := (\texttt{sum}(l,m,n,i,i,i) \parallel \texttt{sum}(l,m,n,i+1)$ então a expressão recursiva $\texttt{sum\_row}(l,m,n,0)$ indica o processo que executa, em 1 *uct*, a soma simultânea dos elementos correspondentes nas linhas m e n, colocando os resultados na linha l. O conjunto $x = \{\overline{sum\_c^{(li0)}}_{10:00}, \overline{sum\_r^{(li1)}}_{10:00}\}_{i \in \omega}$ representa este processo em $\mathbb{D}_\infty^{\rightarrow}$ e consiste no menor ponto fixo para a equação $X_{i+1} = F_{(10)}(x_{i+1} \sqcap x_i)$, onde $x_i = \{\overline{sum\_c^{(li0)}}_{10:00}, \overline{sum\_r^{(li1)}}_{10:00}\}$.
   Considerações semelhantes podem ser obtidas para o processo $\texttt{sum\_}col(l,i,j,k) := (\texttt{sum}(l,l,l,i,j,k) \parallel \texttt{sum\_}col(l+1,i,j,k)$.

3. Considere o processo $\mathbf{Q}_{li}$ recursivamente definido pela expressão
   $\begin{cases} \texttt{sum\_}row(l,0) & := \quad \textbf{skip} \\ \texttt{sum\_}row(l,i) & := \quad \texttt{sum}(l,l,l,i-1,i,i-1); \texttt{sum\_}row(l,i-1) \end{cases}$
   que executa, seqüencialmente e em $(i-1)$ *uct*, o somatório dos correspondentes intervalos posicionados nas primeiras (i) colunas da linha l, colocando o resultado na primeira coluna. A representação gráfica para $\mathbf{Q} = \texttt{sum\_}row(0,2)$ corresponde ao conjunto
   $\{\overline{sum\_c^{(010)}}_{101:00}, \overline{sum\_r^{(011)}}_{101:00}, \overline{sum\_c^{(000)}}_{111:00}, \overline{sum\_r^{(001)}}_{111:00}\} \in \mathbb{D}_\infty^{\rightarrow}$.
   Observações análogas podem ser obtidas para
   $\begin{cases} \texttt{sum\_col}(0,i) & := \quad \textbf{skip} \\ \texttt{sum\_col}(l,i) & := \quad \texttt{sum}(l-1,l,l-1,i,i,i); \texttt{sum\_col}(l-1,i) \end{cases}$

4. O processo $\mathbf{P}$ é definido pela expressão
   $\begin{cases} \texttt{sum\_row}(0,i) & := \quad \textbf{skip} \\ \texttt{sum\_row}(l,i) & := \quad \texttt{sum\_row}(l,i) \parallel \texttt{sum\_row}(l-1,i) \end{cases}$
   executa, simultaneamente, o processo Q nas l-primeiras linhas.

Logo abaixo, outros processos elementares são relacionados e exemplificações envolvendo o produto de intervalos são definidas utilizando os construtores modelados em MGI. Para facilitar a notação, sejam os números reais $\alpha = s[m,j,0] \cdot s[n,k,0]$, $\beta = s[m,j,1] \cdot s[n,k,0]$, $\gamma = s[m,j,1] \cdot s[n,k,0]$ e $\theta = s[m,j,1] \cdot s[n,k,1]$.

$\mathcal{L}(\mathbb{D}_\infty)$	$\mathbb{D}_\infty$
$\bullet\mathtt{p\_2}(l,m,n,i,j,k,u) \equiv (s[l,i,u+2] := \alpha+\beta+\gamma+\theta)$	$\{p\_2_{:00}^{(l,i,u+2)}\}$
$\bullet\mathtt{p\_3}(l,m,n,i,j,k,u) \equiv (s[l,i,u+3] := \alpha-\beta+\gamma+\theta)$	$\{p\_3_{:00}^{(l,i,u+3)}\}$
$\bullet\mathtt{p\_4}(l,m,n,i,j,k,u) \equiv (s[l,i,u+4] := \alpha+\beta-\gamma+\theta)$	$\{p\_4_{:00}^{(l,i,u+4)}\}$
$\bullet\mathtt{p\_5}(l,m,n,i,j,k,u) \equiv (s[l,i,u+5] := \alpha-\beta-\gamma+\theta)$	$\{p\_5_{:00}^{(l,i,u+2)}\}$

1. A execução simultânea dos processos $\mathtt{p\_2}, \mathtt{p\_3}, \mathtt{p\_4}$ e $\mathtt{p\_5}$ é dada por

$$\begin{aligned}\mathtt{p\_par}(l,m,n,i,j,k) \quad \equiv \quad & (\mathtt{p\_2}(l,m,n,i,j,k,2) \parallel \mathtt{p\_3}(l,m,n,i,j,k,2) \parallel \\ & \mathtt{p\_4}(l,m,n,i,j,k,2) \parallel \mathtt{p\_5}(l,m,n,i,j,k,2)).\end{aligned}$$

2. Considerando-se a definição recursiva abaixo, o processo instanciado $Min(l,i,5)$ calcula o mínimo entre um conjunto finito de intervalos localizados em posições sucessivas de memória, colocando o resultado na posição $s[l,i,2]$.
$$\begin{cases} \mathtt{Min}(l,i,0) & := \quad \mathtt{skip} \\ \mathtt{Min}(l,i,u) & := \quad \mathtt{mini}(l,i,u-1); \mathtt{Min}(l,i,u-1) \end{cases}$$
sempre que $\mathtt{mini}(l,i,0) \equiv \mathtt{mini}(l,i,1) \equiv \mathtt{mini}(l,i,2) = \mathtt{skip}$ e $\mathtt{mini}(l,i,u+3) \equiv (s[l,i,u+2] := \mathtt{min}(s[l,i,u+2], s[l,i,u+3]))$.

   De forma análoga, define-se o processo
$$\begin{cases} \mathtt{Max}(l,i,0) & := \quad \mathtt{skip} \\ \mathtt{Max}(l,i,u) & := \quad \mathtt{maxi}(l,i,u-1); \mathtt{Max}(l,i,u-1) \end{cases}$$
sempre que $\mathtt{maxi}(l,i,0) \equiv \mathtt{maxi}(l,i,1) \equiv \mathtt{maxi}(l,i,2) = \mathtt{skip}$ e $\mathtt{maxi}(l,i,u+3) \equiv (s[l,i,u+2] := \mathtt{max}(s[l,i,u+2], s[l,i,u+3]))$.

3. O processo que executa, simultaneamente, o produto dos intervalos correspondentes nas linhas m e n, colocando os resultados na linha l, é dado por $\mathtt{prod\_row}(l,m,n,i) \equiv (\mathtt{p}(l,m,n,i,i,i) \parallel \mathtt{prod\_row}(l,m,n,i+1))$, onde

$$\begin{aligned}\mathtt{p}(l,m,n,i,j,k) \quad \equiv \quad & (\mathtt{p\_par}(l,m,n,i,j,k); \\ & (\mathtt{p\_c}(l,m,n,i,j,k) \parallel \mathtt{p\_r}(l,m,n,i,j,k))), \text{ sendo que} \\ \mathtt{p\_c}(l,m,n,i,j,k) \quad \equiv \quad & (s[l,i,0] := (Max(l,i,5) + Min(l,i,5))/2) \text{ e} \\ \mathtt{p\_r}(l,m,n,i,j,k) \quad \equiv \quad & (s[l,i,1] := (Max(l,i,5) - Min(l,i,5))/2).\end{aligned}$$

5. Conclusões

Utilizando a Teoria dos Domínios, mais precisamente os Espaços Coerentes, este trabalho introduz o modelo intervalar da Máquina Geométrica. A MG é um modelo abstrato, admitindo memória infinita com tempo de acesso à memória constante,

características não compatíveis com a realidade. Entretanto, mostra-se que este modelo de máquina constitui-se num eficiente instrumento para estudo da estrutura lógica e ordenada de computações intervalares paralelas e não-determinísticas. Ou seja, neste contexto, o desenvolvimento de algoritmos para a MGI pode fundamentar o desenvolvimento de algoritmos práticos para computações intervalares.

Abstract. This paper introduces the interval version of the Geometric Machine (GM) Model, to allow the representation of algorithms of Interval Mathematics, where the set of values in the GM memory is represented by the coherence space of rational intervals $\mathbb{IQ}$, a constructive computational representation of the space of real intervals. The GM model is based on coherence spaces. The infinite GM memory, represented by the coherence space $\mathbb{S}$ of states, is conceived as spatially distributed points in a geometric space. Following this approach, a semantic modelling of interval algorithms using interval arithmetic operations is presented.

Referências

[1] G. Birkhoff, Lattice Theory, *Bull. Amer. Math. Soc.*, **25** (1967).

[2] G.P. Dimuro, A.C.R. Costa e D.M. Claudio, A Coherence Space of Rational Intervals for a Construction of IR, *Reliable Computing*, **6**, No. 2 (2000), 139-178.

[3] J.-Y. Girard, Linear Logic, *Theoretical Computer Science*, **1** (1987), 187-212.

[4] V. Stoltenberg-Hansen, I. Lindström e E.R. Griffor, "Mathematical Theory of Domains", Cambridge University Press, Cambridge, 1994.

[5] R. Moore, "Interval Analysis", Prentice-Hall, 1966.

[6] R. Reiser, "Estudo da Categoria Computável dos Espaços Coerentes Gerados por Conjuntos Básicos com Aplicação na Análise Real", PPGC/UFRGS, 1997.

[7] R. Reiser, "The Geometric Machine - a computational model for concurrence and non-determinism based on coherence spaces", PhD Thesis, PPGC/UFRGS, Porto Alegre, 2002. (disponível em http://gmc.ucpel.tche.br/imqd)

[8] R.H.S. Reiser, A.C.R. Costa e G.P. Dimuro, First steps in the construction of the Geometric Machine, *TEMA Tend. Mat. Apl. Comput.*, **3**, No. 1 (2002), 183-192.

[9] R.H.S. Reiser, A.C.R. Costa e G.P. Dimuro, A programming language for the Interval Geometric Machine, *Elet. Notes in Theor. Comp. Sci.*, **84** (2003), 1-12.

[10] D. Scott, Some definitional suggestions for automata theory, *Journal of Computer and System Sciences*, **1**, No. 1 (1967), 187-212.

[11] D. Scott, The lattice of flow diagrams, *Lecture Notes in Mathematics*, **188** (1971), 311-372.

[12] A.S. Troeltra, Lectures on Linear Logic, *CSLI Lecture Notes*, **29** (1992).