2
Teoria dos Problemas – Visão Categórica

Neste capítulo, apresentamos as definições básicas utilizadas no restante do trabalho. Em especial, examinamos várias definições alternativas de categorias de problemas e reduções, e argumentamos por que a alternativa adotada no trabalho não acarreta perda de generalidade para nossos propósitos.

2.1
Problemas e Reduções

2.1.1 Definição: problema. Um problema é especificado por um conjunto de instâncias, um conjunto de resultados possíveis e uma relação que determina quais resultados são corretos para quais instâncias.

Mais precisamente, um problema é uma tripla $\langle\, D, R, p \,\rangle$, com D e R conjuntos não-vazios, e $p \subseteq D \times R$ uma relação, $p \neq \varnothing$. Os elementos $d \in D$ são chamados de *dados* ou *instâncias*; os elementos $r \in R$ são chamados de *resultados* ou *respostas*; a relação p é chamada de *condição do problema*; $(d, r) \in p$ significa que r é uma resposta correta para a instância d.

2.1.2 Reduções. Problemas se relacionam através da noção de *redução*. Afirmar que um problema P se reduz a um problema P' é, em termos imprecisos, afirmar que as instâncias de P podem ser transformadas em instâncias de P' de tal forma que, quando respostas corretas para estas instâncias de P' forem encontradas, estas respostas podem ser transformadas em respostas corretas para as instâncias de P.

Diferentes definições de redução surgem quando aceitamos diferentes maneiras de transformar instâncias de P em instâncias de P' e de transformar respostas de

P' em respostas de P. Algumas questões a considerar são

- As transformações devem ser funcionais (determinísticas) ou transformações não-determinísticas são aceitas?
- As transformações devem ser computáveis?
- Caso a resposta ao item anterior seja afirmativa, deve haver um limite para a complexidade computacional (de tempo e/ou de espaço) das transformações?
- Se as transformações são funções, quais são os domínios destas funções?

Os trabalhos originais sobre Teoria Geral dos Problemas (TGP) [26], justamente por objetivarem a generalidade, requeriam apenas que as transformações envolvidas em uma redução fossem de natureza funcional, mas a teoria incluía mecanismos para definir classes mais restritas de funções.

De fato, grande parte das definições encontradas na literatura estipulam que as transformações devem ser funções computáveis em tempo determinístico polinomial ([27, 28, 29]) ou em espaço logarítmico ([30]).

Quanto aos domínios das funções de transformação, parte do trabalho sobre TGP ([31, 32]) foi dedicado ao estudo de aridades diferentes para a função de transformação de respostas. Mais precisamente, dados dois problemas $\langle D, R, p \rangle$ e $\langle D', R', p' \rangle$ tais que P se reduz a P', a função σ de transformação de respostas pode ser de um dos seguintes tipos:

- Uma função unária $\sigma : R' \rightarrow R$, significando que uma resposta r' de P' é transformada em uma resposta de P sem que seja usada qualquer outra informação como argumento.

  Este tipo de função de transformação de respostas é usado, por exemplo, na definição de *reduções de Karp* entre problemas de decisão ([28, 29]). Problemas de decisão são aqueles cujo domínio de respostas é o conjunto $\{S, N\}$, com elementos representando "sim" e "não". Na verdade, em uma redução de Karp, a função de transformações de respostas é sempre a função identidade.

  Funções unárias σ também são usadas em [30] para definir reduções entre problemas de tipos quaisquer, sem a restrição a problemas de decisão.
- Uma função binária $\sigma : D \times R' \rightarrow R$, significando que a transformação de respostas pode fazer uso de informação sobre a instância de P que estava sendo resolvida originalmente.

  Este tipo de função de transformação de respostas está envolvido nas chamadas *reduções de Turing* ([28, 29]). Além disso, dependendo de como a informação sobre a instância original é utilizada pela função de transformação de respostas, diferentes subtipos de reduções de Turing podem ser definidos, como reduções via tabela-verdade ("*truth-table reducibilities*"), etc. Ver [27].

- Uma função binária $\sigma : D' \times R' \to R$, significando que a transformação de respostas tem a sua disposição apenas informação sobre a instância de P' resultante da transformação da instância original de P, além da resposta de R' que deve ser transformada. [32] mostra que, sempre que houver, entre dois problemas dados, uma redução com $\sigma : D' \times R' \to R$, haverá também uma redução, entre os mesmos dois problemas, com $\sigma : D \times R' \to R$.

  Reduções com $\sigma : D' \times R' \to R$ não podem ser compostas no caso geral, o que impede a definição de uma categoria de problemas tendo como morfismos reduções deste tipo. Por isso, funções do tipo $\sigma : D' \times R' \to R$ não serão mais consideradas neste trabalho.

- Uma função ternária $\sigma : D \times D' \times R' \to R$, significando que a transformação de respostas pode fazer uso de informação sobre a instância original de P, sobre a instância de P' resultante da transformação da instância original de P, e sobre a resposta de P' a transformar.

  O mesmo resultado de [32] citado acima significa que, sempre que houver, entre dois problemas dados, uma redução com função ternária de transformação de respostas, haverá também uma redução, entre os mesmos dois problemas, com função binária de transformação de respostas do tipo $\sigma : D \times R' \to R$. Além disso, a recíproca é verdadeira, fazendo com que os pares de problemas relacionados por reduções com funções de transformação de respostas do tipo $\sigma : D \times R' \to R$ coincidam com os pares de problemas relacionados por reduções com funções ternárias de transformação de respostas. Assim, funções ternárias de transformação de respostas não serão mais consideradas neste trabalho.

A seguir, consideramos duas definições diferentes de reduções entre problemas, uma envolvendo funções unárias de transformação de respostas e outra envolvendo funções binárias de transformação de respostas. Note que, em ambos os casos, restringimo-nos a funções computáveis em tempo polinomial.

Na seção 2.4, mais adiante, faremos a opção por uma das definições, apresentando argumentos para justificar nossa escolha.

2.1.3 Definição: Redução Unária. Dados os problemas $\langle\ D, R, p\ \rangle$ e $\langle\ D', R', p'\ \rangle$, uma *redução unária* $P \stackrel{(\tau,\sigma)}{\longrightarrow} P'$ consiste em um par de funções computáveis em tempo polinomial (τ, σ), com $\tau : D \to D'$ e $\sigma : R' \to R$ tais que respostas corretas são preservadas; mais precisamente, para todo $d \in D$ e para todo $r' \in R'$

$$(\tau(d), r') \in p' \quad \Rightarrow \quad (d, \sigma(r')) \in p$$

Além disto, exigimos que instâncias de D que possuam resposta correta sejam mapeadas para instâncias de D' que também possuam resposta correta. Mais precisamente,

$$\forall d \in D : [(\exists r \in R : (d, r) \in p) \Rightarrow (\exists r' \in p' : (\tau(d), r') \in p')]$$

2.1.4 Definição: Redução Binária. Dados os problemas $\langle\ D, R, p\ \rangle$ e $\langle\ D', R', p'\ \rangle$, uma *redução binária* $P \xrightarrow{(\tau,\sigma)} P'$ consiste em um par de funções computáveis em tempo polinomial (τ, σ), com $\tau : D \to D'$ e $\sigma : D \times R' \to R$ tais que respostas corretas são preservadas; mais precisamente, para todo $d \in D$ e para todo $r' \in R'$

$$(\tau(d), r') \in p' \quad \Rightarrow \quad (d, \sigma(d, r')) \in p$$

Como na definição de redução unária, também exigimos que instâncias de D que possuam resposta correta sejam mapeadas para instâncias de D' que possuam resposta correta.

2.1.5 Exemplo: Construção de Circuito Hamiltoniano (CCH). O problema de decisão Circuito Hamiltoniano requer que, dado um grafo direcionado g, respondamos se g possui ou não um circuito hamiltoniano (i.e., um circuito que visita cada nó de g exatamente uma vez). Aqui, definimos uma versão onde, caso g não possua um circuito hamiltoniano, a resposta deve ser "não", e, caso g possua um circuito hamiltoniano, a resposta deve ser o próprio circuito hamiltoniano, em vez de apenas "sim". Chamamos este problema de Construção de Circuito Hamiltoniano. Definimos o problema CCH como

$$\text{CCH} = \langle\ G, C, p_{\text{CCH}}\ \rangle$$

onde

- G é o conjunto de todos os grafos simples direcionados finitos;
- C é o conjunto contendo o elemento N (que significa "não"), além de todos os circuitos (hamiltonianos ou não) dos grafos de G;

– $p_{\mathrm{CCH}} \subseteq G \times C$ é a relação tal que, dados o grafo $g \in G$ e o circuito $c \in C$, $(g, c) \in p_{\mathrm{CCH}}$ se e somente se c é um circuito hamiltoniano em g; além disso, $(g, N) \in p_{\mathrm{CCH}}$ se e somente se g não possui nenhum circuito hamiltoniano.

2.1.6 Exemplo: Problema do Caixeiro Viajante Construtivo (CVC).

Analogamente ao exemplo anterior, definimos aqui uma versão construtiva do problema do caixeiro viajante, onde uma resposta afirmativa não é apenas um "sim", mas um circuito que serve de testemunha para o "sim". Definimos o problema do CVC como

$$\mathrm{CVC} = \langle\ GC \times \mathbb{N}^{\star}, T, p_{\mathrm{CVC}}\ \rangle$$

onde

– GC é o conjunto de todos os grafos direcionados completos finitos com pesos positivos nos arcos;

– $\mathbb{N}^{\star}$ é o conjunto dos naturais maiores que zero;

– T é o conjunto contendo o elemento N (que significa "não"), além de todas as "turnês" (permutações dos nós) dos grafos de GC;

– $p_{\mathrm{CVC}} \subseteq (GC \times \mathbb{N}^{\star}) \times T$ é a relação tal que, dadas a instância $(g, k) \in GC \times \mathbb{N}^{\star}$ e a turnê $t \in T$, $((g, k), t) \in p_{\mathrm{CVC}}$ se e somente se t é uma turnê de g com custo total menor ou igual a k; além disso, $((g, k), N)) \in p_{\mathrm{CVC}}$ se e somente se g não possui nenhuma turnê com custo menor ou igual a k.

2.1.7 Exemplo: uma redução unária CCH → CVC.

$\mathrm{CCH} \xrightarrow{(\tau,\sigma)} \mathrm{CVC}$ através das seguintes funções [28]:

– $\tau : G \to GC \times \mathbb{N}^{\star}$ que mapeia um grafo g com conjunto de nós N e conjunto de arcos E no par $(g^{\star}, |N|)$, onde $g^{\star}$ é o grafo completo com o mesmo conjunto de nós N de g e com o arcos pertencentes a E rotulados com peso 1 e os arcos não-pertencentes a E rotulados com peso 2; $|N|$ é a quantidade de nós no grafo g;

– $\sigma : T \to C$ mapeia cada turnê t no circuito hamiltoniano correspondente, e mapeia N em N.

(τ, σ) é uma redução, pois

1. Toda instância de CCH que possui resposta correta é mapeada por τ em uma instância de CVC que possui resposta correta. (Na verdade, todas as instâncias de ambos os problemas possuem respostas corretas.);

2. Dado um grafo $g \in G$ composto de k nós, se $g^\star$ não possui uma turnê com custo menor ou igual a k, então g não possui um circuito hamiltoniano. Se $g^\star$ possui uma turnê com custo menor ou igual a k, então a sequência de nós da turnê forma um circuito hamiltoniano de g;

3. τ e σ são computáveis em tempo polinomial.

2.1.8 Exemplo: Torres de Hanói (TH). O clássico problema das torres de Hanói com 3 postes A, B e C pode ser descrito como

$$\mathrm{TH} = \langle\ \mathbb{N}, S, p_{\mathrm{TH}}\ \rangle$$

onde

- Uma instância é um natural k representando o número de discos;
- S é o conjunto de todas as sequências finitas formadas pelas instruções
  - `Mover de A para B`;
  - `Mover de A para C`;
  - `Mover de B para C`;
  - `Mover de B para A`;
  - `Mover de C para A`;
  - `Mover de C para B`;
- $(k, s) \in p_{\mathrm{TH}}$ se e somente se s é uma sequência de instruções que, aplicada ao estado em que existem k discos de tamanhos diferentes empilhados no poste A e nenhum disco nos outros dois postes, resulta nos k discos empilhados no poste C (respeitando a condição de que um disco maior nunca pode ser empilhado sobre um disco menor). Em especial, $(0, \varepsilon) \in p_{\mathrm{TH}}$, onde ε é a sequência vazia.

2.1.9 Exemplo: uma auto-redução unária TH → TH. A idéia de que, se tivermos uma resposta para uma instância k de TH, então podemos transformar esta resposta em uma resposta para a instância $k-1$ é expressa na seguinte redução unária $\mathrm{TH} \xrightarrow{(\tau,\sigma)} \mathrm{TH}$:

- $\tau : k \mapsto k + 1$;
- $\sigma : s \mapsto s^\star$, onde $s^\star$ é a sequência que resulta de eliminarmos de s de todas as instruções que movem o menor disco.

2.1.10 Exemplo: problemas de decisão. Problemas de decisão são aqueles da forma $\langle\ D, \{S, N\}, p\ \rangle$, com p uma relação funcional. Reduções de Karp ([28]) são reduções (τ, σ) entre problemas de decisão nas quais $\sigma = \mathrm{id}$.

2.1.11 Exemplo: problemas de otimização. Um problema de otimização é um problema da forma $\langle\ D, R, p\ \rangle$, onde

- Os elementos de D são pares $(d, m_d : R \to \mathbb{N})$, com m_d uma função parcial de avaliação, atribuindo a cada $r \in R$ o valor de r quando r é visto como uma possível resposta para a instância d; se r não for uma resposta viável para d, então $m_d(r)$ é indefinida; [1]
- $p = \{((d, m_d), \bar{r}) \mid \forall r \in R \text{ com } m_d(r) \downarrow\ :\ m_d(\bar{r}) \geq m_d(r)\}$ no caso de um problema de maximização, ou
- $p = \{((d, m_d), \bar{r}) \mid \forall r \in R \text{ com } m_d(r) \downarrow\ :\ m_d(\bar{r}) \leq m_d(r)\}$ no caso de um problema de minimização.

Notação: Para todo par $((d, m_d), \bar{r}) \in p$, usaremos $m_d^\star$ para denotar o valor $m_d(\bar{r})$, i.e., o valor da avaliação da resposta ótima $\bar{r}$ para a instância (d, m_d).

2.1.12 Exemplo: reduções binárias de problemas de decisão para problemas de otimização. Suponha que $\langle\ D, R, p\ \rangle$ é um problema de maximização (a situação é análoga para problemas de minimização). P pode ser associado a um problema de decisão $P_{\mathrm{dec}} = \langle\ D \times \mathbb{N}, \{S, N\}, p_{\mathrm{dec}}\ \rangle$, onde

$$p_{\mathrm{dec}} = \{(d, m_d, k, S) \mid m_d^\star \geq k\}\ \cup\ \{(d, m_d, k, N) \mid m_d^\star < k\}$$

Ou seja, dados uma instância (d, m_d) e um natural k, o problema de decisão responde se $m_d^\star$ (a resposta ótima para (d, m_d)) tem valor maior ou igual a k. Se as

[1] Escrevemos $m_d(r) \downarrow$ para dizer que $m_d(r)$ é definida, e $m_d(r) \uparrow$ para dizer que $m_d(r)$ é indefinida.

funções de avaliação m_d forem computáveis em tempo polinomial para todo d, então existe uma redução $P_{\text{dec}} \xrightarrow{(\tau,\sigma)} P$ com

– $\tau : (d, m_d, k) \mapsto (d, m_d)$;

– $\sigma : (d, m_d, k, r) \mapsto \begin{cases} S & \text{se } m_d(r) \geq k \\ N & \text{se } m_d(r) < k \text{ ou } m_d(r)\uparrow \end{cases}$

Salvo em casos especiais, esta redução não pode ser definida como unária. Mais adiante, porém, na seção 2.4, veremos como transformar reduções binárias em reduções unárias.

2.1.13 Exemplo: algoritmos recursivos como auto-reduções binárias. Considere o problema de computar o comprimento de uma lista de elementos de um tipo qualquer. Este problema pode ser representado por $C = \langle\ L, \mathbb{N}, p\ \rangle$, onde L é o conjunto de listas em questão (com **nil** denotando a lista vazia) e $p = \{(s,k) \mid |s| = k\}$. A seguinte auto-redução binária $C \xrightarrow{(\tau,\sigma)} C$ equivale ao conhecido algoritmo recursivo para computar o comprimento de uma lista: dada uma lista s, retornar 0 se $s = \mathbf{nil}$; caso contrário, retornar $1 + |\text{cdr}(s)|$.

– $\tau : s \mapsto \begin{cases} \mathbf{nil} & \text{se } s = \mathbf{nil} \\ \text{cdr}(s) & \text{se } s \neq \mathbf{nil} \end{cases}$

– $\sigma : (s, k) \mapsto \begin{cases} 0 & \text{se } s = \mathbf{nil} \\ k+1 & \text{se } s \neq \mathbf{nil} \end{cases}$

O algoritmo recursivo equivalente a esta redução não pode ser representado por uma redução unária. Mais adiante, porém, na seção 2.4, veremos como transformar reduções binárias em reduções unárias.

2.2
Categorias de Problemas

2.2.1 Universos de problemas. Na definição de problema em 2.1.1, não foi especificado o universo habitado pelos conjuntos de instâncias e respostas. Se permitirmos que qualquer conjunto sirva como D ou R na definição de um problema, logo nos depararemos com problemas de fundamentação. Mais adiante, (na prova do Teorema 3.4.8, por exemplo), precisaremos do fato de que a coleção de todos os problemas forma um conjunto.[2]

Para resolver esta questão, assumimos a existência de um universo $\mathcal{U}$ (cujos elementos são os conjuntos de dados e os conjuntos de respostas dos nossos problemas) tal que *a coleção de todos os conjuntos de instâncias e todos os conjuntos de respostas forme um conjunto também*. Em outras palavras, $\mathcal{U}$ é um conjunto.

Mais ainda: exigiremos que cada instância e cada resposta nos seja dada segundo algum esquema de codificação, como cadeias finitas de 0s e 1s, por exemplo (que é o que ocorre na prática). Ver [28, p. 21] para um bom exemplo de um esquema de codificação. Assim, apenas conjuntos enumeráveis poderão ser aceitos como D ou R na definição de problema.

2.2.2 Universos alternativos. Contanto que a escolha do universo preserve a verdade dos resultados principais desta tese, podemos permitir universos diferentes a partir dos quais problemas podem ser construídos. De fato, isto será feito no Capítulo 6, quando desejaremos tratar apenas de problemas onde as respostas são, elas mesmas, conjuntos.

Esta liberdade de escolha permite a construção de um modelo parametrizado pelo universo de dados e respostas dos problemas.

2.2.3 Exemplo de universo. Seja Σ um alfabeto finito adequado.[3] Por exemplo, $\Sigma = \{0, 1\}$. Podemos definir um universo $\mathcal{U} = \wp^{+}(\Sigma^{\star})$, cujos elementos são todos os conjuntos não-vazios de cadeias finitas formadas com os símbolos de Σ. Assim, dado um problema $\langle\ D, R, p\ \rangle$, temos que $D, R \in \mathcal{U}$ são conjuntos não-vazios de cadeias.

Um esquema de codificação associa a cada cadeia em $\Sigma^{\star}$ um "significado", i.e., um objeto matemático (por exemplo, um grafo, um conjunto, uma função) do qual a cadeia é vista como uma representação.

---

[2]Uma categoria é dita *pequena* quando a coleção de seus objetos é um conjunto; caso contrário, a categoria é dita *grande*. A categoria **Set**, cujos objetos são todos os conjuntos, é uma categoria grande. Se qualquer conjunto fosse aceito como D ou R na definição de problema, a categoria dos problemas também seria grande.

[3]Para efeito do cálculo da complexidade de algoritmos, não é considerado adequado, por exemplo, um alfabeto contendo apenas um símbolo. Ver detalhes em [28].

2.2.4 Definição: a categoria $\mathbf{Prob}_1$ de problemas e reduções unárias. Definimos a categoria $\mathbf{Prob}_1$, cujos objetos são problemas e cujos morfismos são reduções unárias. Claramente, para cada problema $\langle\ D, R, p\ \rangle$ existe uma redução identidade $(\mathrm{id}_D, \mathrm{id}_R)$; dadas duas reduções (τ_1, σ_1) e (τ_2, σ_2), a redução composta é $(\tau_2 \circ \tau_1, \sigma_1 \circ \sigma_2)$, e a composição de reduções é associativa.

2.2.5 Definição: a categoria $\mathbf{Prob}_2$ de problemas e reduções binárias. Definimos a categoria $\mathbf{Prob}_2$, cujos objetos são problemas e cujos morfismos são reduções binárias. Dado um problema $\langle\ D, R, p\ \rangle$, a redução identidade é dada por (id_D, π_2), onde π_2 é a segunda projeção $D \times R \to R$. Dadas duas reduções

$$\langle\ D_1, R_1, p_1\ \rangle \stackrel{(\tau_1,\sigma_1)}{\longrightarrow} \langle\ D_2, R_2, p_2\ \rangle$$

e

$$\langle\ D_2, R_2, p_2\ \rangle \stackrel{(\tau_2,\sigma_2)}{\longrightarrow} \langle\ D_3, R_3, p_3\ \rangle$$

a composta é dada por (τ, σ), onde

$$\begin{array}{rrcl} \tau: & D_1 & \to & D_3 \\ & d_1 & \mapsto & \tau_2(\tau_1(d_1)) \end{array}$$

e

$$\begin{array}{rrcl} \sigma: & D_1 \times R_3 & \to & R_1 \\ & (d_1, r_3) & \mapsto & \sigma_1(d_1, \sigma_2(\tau_1(d_1), r_3)) \end{array}$$

A composição de reduções assim definida é associativa.

2.2.6 Categorias pequenas. Ambas as categorias definidas acima são pequenas; i.e., suas coleções de objetos formam conjuntos. Isto decorre da escolha de universos de problemas descrita em 2.2.1. Este fato será importante mais adiante, na prova do teorema 3.4.8, quando mostraremos que uma determinada categoria construída a partir de $\mathbf{Prob}_1$ é um topos.

2.3
Problemas Solúveis × Parcialmente Solúveis

2.3.1 Definição. Um problema $\langle\, D, R, p \,\rangle$ é dito *solúvel* (ou *totalmente solúvel*) se toda instância possui pelo menos uma resposta correta; i.e., para todo $d \in D$ existe $r \in R$ tal que $(d, r) \in p$. Caso contrário, o problema é dito *parcialmente solúvel.*[4]

2.3.2 Transformando problemas parcialmente solúveis em solúveis. A cada problema parcialmente solúvel $\langle\, D, R, p \,\rangle$, parece razoável associar um problema totalmente solúvel $P_S = \langle\, D_S, R, p \,\rangle$ cujo conjunto de instâncias contém exatamente as instâncias solúveis de P. (Por nossas definições, pelo menos uma instância de P deve ser solúvel.)

Esta idéia produz um funtor S de $\mathbf{Prob}_2$ para uma subcategoria plena $\mathbf{Prob}_2^S$ de $\mathbf{Prob}_2$, conforme definido abaixo. A seguir, mostramos que este funtor S é parte de uma adjunção interessante, que faz de $\mathbf{Prob}_2^S$ uma subcategoria co-reflectiva de $\mathbf{Prob}_2$.

2.3.3 Definição: $\mathbf{Prob}_2^S$ e o funtor S. Seja $\mathbf{Prob}_2^S$ a subcategoria plena de $\mathbf{Prob}_2$ cujos objetos são exatamente os problemas totalmente solúveis. Definimos o funtor $S : \mathbf{Prob}_2 \to \mathbf{Prob}_2^S$ que mapeia cada problema $\langle\, D, R, p \,\rangle$ no problema totalmente solúvel

$$P_S = \langle\, D_S, R, p \,\rangle$$

onde $D_S = \{d \in D \mid \exists r \in R : (d, r) \in p\}$.

S mapeia cada redução $P \xrightarrow{(\tau,\sigma)} P'$ na redução

$$\langle\, D_S, R, p \,\rangle \xrightarrow{(\tau_S,\sigma)} \langle\, D'_S, R', p' \,\rangle$$

onde τ_S é a restrição de τ a D_S.

2.3.4 Teorema: $\mathbf{Prob}_2^S$ é uma subcategoria co-reflectiva de $\mathbf{Prob}_2$. [5]

2.3.5 Observações.

[4] Note que problemas *totalmente insolúveis* (i.e., com $p = \varnothing$) não são admitidos em nossa definição – ver 2.1.1.

[5] Provas detalhadas das proposições e dos teoremas enunciados no texto desta tese encontram-se no apêndice A.

– O mesmo raciocínio pode ser aplicado à categoria $\mathbf{Prob}_1$ de problemas e reduções unárias, produzindo a categoria $\mathbf{Prob}_1^S$ de problemas solúveis e reduções unárias.

– Em $\mathbf{Prob}_1^S$ e $\mathbf{Prob}_2^S$, como todos os problemas são solúveis, as reduções sempre mapeiam instâncias solúveis em instâncias solúveis, tornando desnecessárias as condições correspondentes nas definições de redução (2.1.3 e 2.1.4).

– Dado o teorema 2.3.4, passamos a trabalhar, deste momento em diante, apenas com as categorias $\mathbf{Prob}_1^S$ e $\mathbf{Prob}_2^S$.

2.4
Reduções Unárias × Reduções Binárias

2.4.1 Trabalhando apenas com reduções unárias. Esta seção apresenta argumentos para justificar nossa decisão de considerar apenas reduções unárias entre problemas. Isto se deve ao fato de que é possível trabalhar em $\mathbf{Prob}_1^S$ e ainda assim fazer referência a reduções binárias. Mais precisamente, demonstraremos que (1) existe uma imersão de $\mathbf{Prob}_2^S$ em $\mathbf{Prob}_1^S$, e que (2) $\mathbf{Prob}_1^S$ é uma subcategoria reflectiva de $\mathbf{Prob}_2^S$.

2.4.2 Transformando reduções binárias em reduções unárias. A idéia principal é a de que um problema $\langle\, D, R, p\,\rangle$ pode ser associado a um problema diferente $\bar{P} = \langle\, D, \mathrm{poly}(R^D), \bar{p}\,\rangle$, onde as respostas possíveis, agora, são todas as funções de D para R computáveis em tempo polinomial. Cada uma destas funções pode ser vista como um programa que tenta resolver as instâncias em D. Neste novo problema $\bar{P}$, consideramos que uma função f é uma resposta correta para a instância d se e somente se o valor $f(d)$ for uma resposta correta para d no problema original P. (Observe que os valores atribuídos por f a instâncias diferentes de d não importam.)

Estendemos esta associação a reduções, mapeando uma redução binária (τ, σ) em uma redução unária $(\tau, \bar{\sigma})$ de acordo com a definição abaixo:

2.4.3 Definição: o mapeamento U. Seja $U : \mathbf{Prob}_2^S \to \mathbf{Prob}_1^S$ o mapeamento que associa cada problema $\langle\, D, R, p\,\rangle$ ao problema

$$\bar{P} = \langle\, D, \mathrm{poly}(R^D), \bar{p}\,\rangle$$

onde

- $\text{poly}(R^D)$ é o conjunto de todas as funções computáveis em tempo polinomial de D para R;
- $\bar{p} = \{(d, f) \mid (d, f(d)) \in p\}$.

O mapeamento U associa cada redução $P \xrightarrow{(\tau,\sigma)} P'$ à redução

$$\bar{P} \xrightarrow{(\tau,\bar{\sigma})} \bar{P'}$$

onde

$$\begin{array}{rlcl} \bar{\sigma}: & \text{poly}(R'^{D'}) & \to & \text{poly}(R^D) \\ & f' & \mapsto & \sigma \circ \langle \, \text{id}_D, f' \circ \tau \, \rangle \end{array}$$

i.e. $\bar{\sigma}(f') : D \to R$ é a função que mapeia cada $d \in D$ em $\sigma(d, f'(\tau(d)))$.

2.4.4 Observações. Note que $\bar{\sigma}$ é uma função computável em tempo polinomial, pois dada uma (codificação de uma) função f' como entrada, a ação de $\bar{\sigma}$ é apenas a de "acrescentar código" a f' para chamar as funções τ e σ com os argumentos apropriados. Como a quantidade de código acrescentado não depende de f', a função $\bar{\sigma}$, na verdade, é computável *em tempo linear*.

Que $(\tau, \bar{\sigma})$ é uma redução em $\mathbf{Prob}_1^S$ advém do fato de que, para todo $d \in D$ e para todo $f' \in \text{poly}(R'^{D'})$, se

$$(\tau(d), f') \in \bar{p'}$$

então

$$(\tau(d), f'(\tau(d))) \in p'$$

o que, por sua vez (por causa da redução original em $\mathbf{Prob}_2^S$), implica que

$$(d, \sigma(d, f'(\tau(d)))) \in p$$

Logo, pela definição de $\bar{\sigma}$, temos que

$$(d, \bar{\sigma}(f')) \in \bar{p}$$

2.4.5 Proposição: U é um funtor de $\mathbf{Prob}_2^S$ para $\mathbf{Prob}_1^S$.

2.4.6 Teorema: o funtor U é uma imersão (não-plena) de $\mathbf{Prob}_2^S$ em $\mathbf{Prob}_1^S$.

2.4.7 A imersão de $\mathbf{Prob}_2^S$ em $\mathbf{Prob}_1^S$ não é plena. Como observado em 2.4.4, uma redução $P \xrightarrow{(\tau,\sigma)} P'$ é mapeada pelo funtor U para uma redução $UP \xrightarrow{(\tau,\bar{\sigma})} UP'$ cuja função de transformação de respostas $\bar{\sigma}$ é computável em tempo linear. Certamente, de UP para UP' existem outras reduções cujas funções de transfomação de respostas são computáveis em tempo polinomial não-linear; estas outras reduções não estão na imagem do funtor U; logo, a imersão U não é plena (*full*).

2.4.8 Reduções unárias podem ser vistas como binárias. $\mathbf{Prob}_1^S$ é isomorfa a uma subcategoria ampla (*wide*) e não-plena (*non-full*) de $\mathbf{Prob}_2^S$, pois toda função unária pode ser vista como uma função binária que não depende de um dos argumentos:

2.4.9 Proposição: $\mathbf{Prob}_1^S$ é (isomorfa a) uma subcategoria de $\mathbf{Prob}_2^S$.

2.4.10 Teorema: $\mathbf{Prob}_1^S$ é (isomorfa a) uma subcategoria reflectiva de $\mathbf{Prob}_2^S$.

2.5 Conclusões do Capítulo

2.5.1 O significado destas subcategorias (co-)reflectivas. Há uma forte semelhança entre nosso trabalho com categorias de problemas e o exemplo mencionado em 1.2.3 envolvendo categorias de AFDs. De início, nosso funtor S, que transforma problemas quaisquer em problemas solúveis, gerando uma subcategoria co-reflectiva, é análogo ao funtor E que elimina estados não-alcançáveis de um AFD. S também corresponde à escolha de subobjetos interessantes de um problema.

Já o nosso funtor $U : \mathbf{Prob}_2^S \to \mathbf{Prob}_1^S$ difere do funtor M minimizador de AFDs, do exemplo em 1.2.3, em alguns aspectos:

– $\mathbf{Prob}_1^S$ possui os mesmos objetos que $\mathbf{Prob}_2^S$;

- O funtor U não é sobrejetivo nos objetos, ao contrário do funtor M;
- Ao contrário do funtor M, o funtor U é injetivo nos objetos; mais ainda: U é uma imersão, criando uma situação interessante, em que a categoria $\mathbf{Prob}_2^S$ é imersível em uma subcategoria reflectiva sua, $\mathbf{Prob}_1^S$;
- Diferentemente do exemplo envolvendo AFDs, a subcategoria reflectiva $\mathbf{Prob}_1^S$ não é plena (*full*).

O funtor U, na verdade, efetua uma modificação nos objetos de $\mathbf{Prob}_2^S$ para permitir a existência de morfismos de um tipo especial (i.e. reduções unárias) entre os objetos modificados.

Apesar das diferenças, a motivação é semelhante àquela do exemplo dos AFDs: julgamos que as informações relevantes de $\mathbf{Prob}_2^S$ ainda estão presentes em sua subcategoria reflectiva $\mathbf{Prob}_1^S$. Isto justifica a definição a seguir:

2.5.2 Definição: a categoria Prob. Este capítulo discutiu alguns aspectos de uma visão categórica de Teoria Geral dos Problemas. A decisão mais importante para o restante do trabalho é a definição da categoria **Prob** de problemas. De agora em diante, trabalharemos com $\mathbf{Prob} = \mathbf{Prob}_1^S$. Em outras palavras:

- Consideraremos apenas problemas totalmente solúveis como objetos de **Prob**;
- Consideraremos apenas reduções unárias como morfismos de **Prob**.

2.5.3 Adequação das definições. As adjunções exibidas neste capítulo fornecem respaldo formal à nossa definição de **Prob**, fazendo-nos crer que os aspectos interessantes de problemas e reduções foram capturados de forma adequada.

Em particular, problemas de decisão (aqueles com $R = \{S, N\}$ e p uma relação funcional) e reduções de Karp formam uma pequena parte de **Prob**. Problemas de decisão são solúveis, e reduções de Karp são unárias (na verdade, a função de transformação de respostas é a identidade). Isto faz com que a discussão sobre problemas totalmente solúveis e insolúveis e a discussão sobre reduções unárias e binárias sejam irrelevantes quando o escopo se limita a problemas de decisão.

Quanto a problemas mais gerais (de busca, de otimização etc.), nossas definições concordam em parte com as de [29, pp. 19, 257], que exigem que uma redução mapeie instâncias solúveis em instâncias solúveis, e com as de [30, p. 229], que admitem apenas funções unárias de transformação de respostas (sem qualquer justificativa, porém). Os exemplos exibidos ao longo deste trabalho são evidência – "empírica", por assim dizer – da adequação das nossas definições.