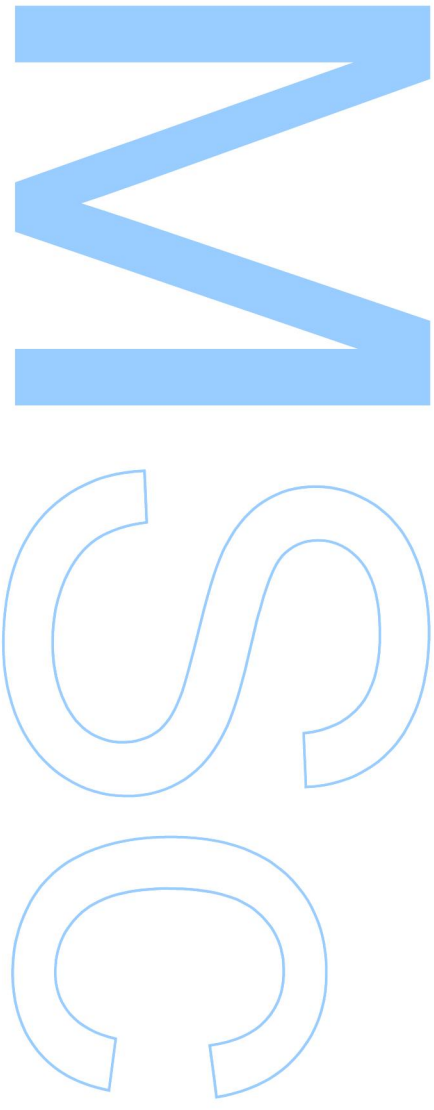
Geração automática de exercícios de trigonometria

Armando Manuel Martins
Mestrado em Ciência de Computadores
Departamento de Ciência de Computadores
2022

Orientador
Ana Paula Tomás
Professora Associada
Faculdade de Ciências da Universidade do Porto





Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, / /

Declaração de honra

Eu, Armando Manuel Ferreira da Silva Martins, inscrito no Mestrado em Ciência dos Computadores da Faculdade de Ciências da Universidade do Porto declaro, nos termos do disposto na alínea a) do artigo 14.º do Código Ético de Conduta Académica da U.Porto, que o conteúdo da presente dissertação de tese reflete as perspetivas, o trabalho de investigação e as minhas interpretações no momento da sua entrega.

Ao entregar esta dissertação de tese, declaro, ainda, que a mesma é resultado do meu próprio trabalho de investigação e contém contributos que não foram utilizados previamente noutros trabalhos apresentados a esta ou outra instituição.

Mais declaro que todas as referências a outros autores respeitam escrupulosamente as regras da atribuição, encontrando-se devidamente citadas no corpo do texto e identificadas na secção de referências bibliográficas. Não são divulgados na presente dissertação de tese quaisquer conteúdos cuja reprodução esteja vedada por direitos de autor.

Tenho consciência de que a prática de plágio e auto-plágio constitui um ilícito académico.

Armando Manuel Ferreira da Silva Martins

Porto, 17 de Outubro de 2022

Abstract

At a time that digital technologies are constantly improving, it is important to provide these advances in the form of tools in the areas that need it most. One of these areas is the teaching of Mathematics. Joining teaching with technology allows the creation of interactive worksheets that contribute to the consolidation of knowledge on a specific topic.

In order to support teachers and students, the AGILMAT project was developed. This core system is an application of constraint logic programming and generates and solves some math exercises, up to the portuguese 12th grade. It is possible to customize the generator through constraints on some parameters, giving some control of the system to the user. The types of exercises generated by the part of prototype we focus on this thesis involve the computation of domains and the solution of equation/inequations.

This dissertation presents the introduction of the module for generating and solving exercises about trigonometry in the AGILMAT prototype. In addition to the pedagogical interest, this module is also relevant in the field of mathematics and computing since methods that operate periodic sets were created. Naturally, a solution to a problem about trigonometry involves operations on periodic and non-periodic sets, such as intersection, for example. Although some of the methods are general, they are intended to be used within the limits imposed by the exercises.

In order to align the student/teacher's expectations with the results of the prototype, the ways in which the exercises and solutions of this topic are presented in high school manuals were studied. We conclude that, very often, the exercises have finite intervals to constrain the general solution (still with an associated period). The generation of exercises follows exactly this structure, an equation/inequation and an interval where the student has to find the solution.

Resumo

Numa altura em que as tecnologias digitais estão em constante melhoria, é importante disponibilizar estes avanços em forma de ferramentas nas áreas que mais precisam. Uma destas áreas é o ensino da Matemática. Juntar o ensino com tecnologia permite a criação de fichas de trabalho interativas que contribuem para a consolidação de conhecimentos num tema específico.

Com o objetivo de apoiar professores e alunos, foi desenvolvido o projeto AGILMAT. O núcleo deste sistema é uma aplicação de programação em lógica por restrições e permite gerar e resolver alguns exercícios de matemática até ao 12º ano escolar. As restrições tomam um papel central na geração porque permitem ao utilizador condicionar a geração para certos requisitos. Os tipos de exercícios gerados pelas componentes em que trabalhámos incidem sobre cálculo de domínios e resolução de equações/inequações.

Nesta dissertação, apresenta-se a introdução de um módulo para geração e resolução de exercícios de trigonometria no protótipo AGILMAT. Para além do interesse pedagógico, este módulo é também relevante no âmbito da matemática e computação uma vez que foram criados métodos que operam conjuntos periódicos. Naturalmente, uma solução de um problema de trigonometria passa por efetuar operações com conjuntos periódicos e não-periódicos, como interseção, por exemplo. Apesar de alguns dos métodos serem gerais, têm o intuito de ser usados dentro dos limites impostos pelos exercícios.

De forma a alinhar a expectativa do aluno/professor com os resultados do gerador, foram estudadas formas comuns de exercícios e soluções sobre esta matéria em manuais do ensino secundário. Concluímos que os exercícios têm muito frequentemente intervalos finitos para restringir a solução geral (ainda com um período associado). A geração de exercícios segue exatamente esta estrutura, uma equação/inequação e um intervalo para o aluno ter de enquadrar a solução que obteve naquele intervalo específico.

Agradecimentos

Estou eternamente grato por todo o apoio que a professora Ana Paula Tomás me deu, dentro e fora do âmbito da dissertação. Pela disponibilidade, delicadeza e profissionalismo. Posso afirmar com toda a certeza que foi decisiva no meu caminho profissional e como ser humano.

Agradeço aos meus pais todo o apoio, educação e sacrifícios que fazem e fizeram por mim.

Incluo os meus amigos que também me ajudaram a ver a luz quando tudo parecia mais escuro.

Dois últimos agradecimentos especiais: ao Paulo Ramos e Alexandra Ferreira por me proporcionarem as melhores condições para a realização desta dissertação.

Dedico esta dissertação ao meu pai e à minha mãe, por todo o sacrifício que fizeram e que fazem por mim.

Conteúdo

Abstract i

Resumo iii

Agradecimentos v

Conteúdo ix

Lista de Figuras xii

Acrónimos xiii

1 Introdução 1

1.1 Desenvolvimento do trabalho e estrutura da dissertação 5

2 Trabalho Relacionado 7

2.1 Tecnologias Digitais no Ensino 7

2.1.1 Motores de Resposta e Sistemas Computacionais Algébricos 7

2.1.2 Cursos *Online* 8

2.1.3 Sistemas de Apoio à Aprendizagem e Avaliação 9

2.2 AGILMAT 9

3 Suporte de conjuntos periódicos 13

3.1 Conjuntos periódicos em forma standard 14

3.1.1 Representação simbólica . . . 15
3.2 Operação de expansão . . . 15
3.3 Operação de interseção . . . 16
3.3.1 Interseção entre conjuntos não-periódicos . . . 17
3.3.2 Interseção entre conjuntos periódicos . . . 18
3.3.3 Interseção entre conjuntos periódicos e não periódicos . . . 18
3.4 Operação de união . . . 19
3.4.1 União de conjuntos não-periódicos . . . 19
3.4.2 União entre conjuntos periódicos . . . 22
3.5 Operação de diferença . . . 22
3.5.1 Diferença entre conjuntos não-periódicos . . . 22
3.5.2 Diferença entre um conjunto não periódico e um conjunto periódico . . . 25

4 Geração de Questões de Trigonometria 27

4.1 Tipos e instâncias de expressões . . . 27
4.2 Extensão para incluir exercícios de trigonometria . . . 28
4.2.1 Extensão da gramática . . . 29
4.3 Construção de Exercícios . . . 32
4.3.1 Geração de Expressões . . . 33
4.3.2 De expressões para exercícios . . . 36

5 Resolução de problemas 41

5.1 Cálculo de domínios . . . 41
5.2 Resolução de equações e inequações . . . 43
5.2.1 Funções trigonométricas simples . . . 44
5.2.2 Somas de Funções Trigonométricas . . . 48

5.2.3 Variantes 49

5.3 Limites do resolutor 51

6 Conclusão **53**

Bibliografia **57**

Lista de Figuras

1.1 Taxas de insucesso em Matemática em Portugal segundo o relatório PISA 2018. 2

1.2 Perguntas de escolha múltipla geradas pelo protótipo AgilmatQuiz (fonte [35]). 4

1.3 Tipos de questões de trigonometria a suportar. 4

2.1 Interface do AGILMAT (fonte [35]). 10

2.2 Visão geral da arquitetura do sistema AGILMAT (fonte [38]). 11

2.3 Funcionamento do núcleo do sistema AGILMAT (traduzido da apresentação em [38]) . 11

2.4 Passos do gerador de exercícios na opção de resolução de restrições, onde E representa uma expressão do ficheiro de expressões. 12

4.1 Excerto da gramática atualizada 31

4.2 Categoria *vtrig* em árvore, sendo f e g são funções polinomiais de grau um. 31

4.3 Representações para tipos e para expressões implementadas na extensão, sendo $f \in \{\sin, \cos, \tan\}$, $trig \in \{\texttt{sin}, \texttt{cos}, \texttt{tan}\}$, F e G combinação de `sin` e `cos` ou ambos `tan`, F' e G' combinação sem repetição de `sin` com `cos`, P e Q do tipo `p1`, $a, b \in \{1, -1\}$, $a', b' \in \mathbb{R}$ e $N \in \mathbb{N}$. 32

4.4 Exemplo de níveis de dificuldade atribuídos a alguns tipos. 34

4.5 Implementação da geração da categoria *vtrig*. 35

4.6 Excerto do predicado `expr/3` para o tipo `sumtrig(_,_)` 35

4.7 Lista de pares tipo-constante, sendo $trig \in \{\texttt{sin}, \texttt{cos}, \texttt{tan}\}$ e A e B os coeficientes de `f` e `g`, respetivamente. 37

5.1 Análise de $\sin x \trianglelefteq \sin k'$ no círculo trigonométrico, quando operador $\trianglelefteq$ é $\geq$ ou é $\leq$. . . 45

5.2 Exemplo do comportamento de $\sin(x)$ no intervalo $]-2\pi, 2\pi[$. 46

5.3 Análise de $\tan x \trianglelefteq \tan k'$ no círculo trigonométrico, quando o operador $\trianglelefteq$ é $>$ ou $<$. . 47

5.4 Análise da restrição $\tan(x) = 1$. 47

5.5 Comportamento da função $\sqrt{\sin(x)}$ no intervalo $]-2\pi, 2\pi[$. 50

Acrónimos

AE *Answer Engines*

ALAS *Assisted Learning and Assessment Systems*

CAS *Computer Algebraic Systems*

CLP *Constraint Logic Programming*

FCUP Faculdade de Ciências da Universidade do Porto

Capítulo 1

Introdução

Ao longo dos últimos vinte anos tem havido um esforço substancial para desenvolver *software* de suporte ao ensino, nomeadamente na área da Matemática. A crescente prevalência das Tecnologias da Informação e da Comunicação (TIC) e, em particular, da Internet (*Web*), tem incentivado o aumento do número de aplicações no ensino assistido [5, 7, 29, 30, 46].

A avaliação e o *feedback* constituem uma parte significativa do trabalho dos professores e dos alunos uma vez que são as métricas indicativas do grau de conhecimento dos alunos. A testagem regular de conhecimentos permite que os professores consigam acompanhar o ritmo de aprendizagem dos alunos. Por sua vez, o *feedback* indica ao aluno o que fazer de acordo com a sua situação.

A Matemática é uma disciplina que requer experiência. Em áreas como esta é fundamental poder dispor de ferramentas para geração automática de exercícios para criar novas oportunidades de aprendizagem. A geração deve produzir exercícios pedagogicamente relevantes, sempre com *feedback* automático imediato. Já os exercícios devem servir para estudo, avaliação formativa (auto-avaliação) e/ou sumativa.

De acordo com o relatório da avaliação realizada em 2018 pelo programa PISA (Programme for International Student Assessment) [17, 23], que avalia grupos de alunos de 15 anos de diversos países/economias, o desempenho em Matemática em Portugal tem vindo a melhorar, embora se verifique uma taxa de insucesso acima dos 20%, como mostra a Figura 1.1. Para combater este defeito, poderá ser útil reforçar a integração de novas metodologias no ensino, com recurso a tecnologia, que ajudem os alunos a compreender conteúdos e procedimentos matemáticos.

Sobre o uso de tecnologia no ensino de Matemática no ensino secundário, importa referir que a equipa que definiu o *Programa e Metas Curriculares de Matemática A*, de 2013, defendeu que a utilização de tecnologia devia ser criteriosa para não condicionar e comprometer gravemente a aprendizagem e a avaliação [3]. A mesma considerou que "apenas a memorização e a compreensão cumulativa

Figura 1.1: Taxas de insucesso em Matemática em Portugal segundo o relatório PISA 2018.

Ano	Posição	Número de participantes	Proporção
2003	3	22	30,1%
2006	5	29	30,9%
2009	10	29	23,8%
2012	10	30	24,9%
2015	11	31	23.8%
2018	13	31	23,3%

de conceitos, técnicas e relações matemáticas permitem alcançar conhecimentos progressivamente mais complexos e resolver problemas progressivamente mais exigentes" [3].

Um estudo realizado em 2015/2016 sobre o uso de computadores nas aulas de Matemática [26], em que participaram 119 professores, concluiu que, apesar de usarem o computador para planear e construir materiais para utilização no contexto de sala de aula, não o usavam para atividades de ensino com os alunos. Isto acontece por diversas razões apontadas na investigação, mas salientamos uma das conclusões: a tecnologia em si não estava devidamente integrada no ensino e por isso existia falta de motivação por parte dos professores para a usar [26].

No entanto, a pandemia por COVID-19 "mudou radicalmente o papel e a perceção da digitalização na sociedade, na economia e, em especial, na educação, acelerou o seu ritmo e as tecnologias tornaram-se essenciais em todas as atividades" [39], como afirma um relatório de 2021 de uma auditoria do Tribunal de Contas. E, acrescenta que "o recurso ao Ensino a Distância em substituição das atividades educativas e letivas presenciais foi a medida mais impactante no âmbito da Educação para mitigar os efeitos da pandemia", nos anos letivos 2019/20 e 2020/21, em Portugal, mas "limitada pela insuficiência de competências e meios digitais" [39].

O *Plano Nacional para a Transição Digital* [25], publicado em 2020 pelo Governo Português, define a Educação Digital como área estratégica de intervenção para a transição digital, no âmbito do "Programa de Digitalização para as Escolas". Das medidas contempladas pelo programa destacamos:

- O acesso a recursos educativos digitais de qualidade (a título de exemplo, manuais escolares, cadernos de atividades, aulas interativas, testes interativos, preparação para exames, análise de desempenho, diagnóstico e proposta de percursos de aprendizagem, relatório de progresso para encarregados de educação e dicionários);
- A definição de processos conducentes à realização e classificação eletrónica de provas de avaliação externa em ambiente digital.

Apesar da relevância dos manuais escolares, a sua estaticidade e o nível de interatividade baixo conduzem a uma experiência limitante. Quando o aluno tem pouco interesse no tema lecionado, torna-se difícil cativá-lo uma vez que o manual está a competir com as tecnologias com acesso à Internet que hoje a maior parte dos jovens a partir dos 16 anos têm no seu bolso [24]. Por outro lado, quando o aluno termina os exercícios sobre um tópico, mas sente que ainda não consolidou bem a matéria, pode apenas voltar a repetir os mesmos exercícios (dos manuais ou fichas produzidas pelos professores) ou procurar exercícios *online*.

A repetição pode trazer vantagens ao aluno, permitindo aprender a aplicação de uma regra, por exemplo. No entanto, a repetição sucessiva pode conduzir à memorização de uma resolução particular, sem uma compreensão da matéria que lhe permita resolver outros problemas cujo padrão não seja igual. Se o aluno procurar exercícios *online* sem supervisão, certamente vai encontrar o tema que procura, mas os exercícios podem não ser adequados ao seu nível. Esta discrepância de dificuldade pode levá-lo a desmotivar ou a tentar resolver os exercícios que encontra, por vezes sem compreender se as suas soluções estão corretas ou perceber erradamente como chegar a uma determinada solução.

Esta dissertação enquadra-se numa linha de investigação iniciada com o projeto AGILMAT (*Automatic Generation of Interactive Mathematics Drills*, POSI/CHS/48565/2002), em que se explorou a aplicação de programação em lógica com restrições (*Constraint Logic Programming* (CLP)) para geração de exercícios de matemática, principalmente no ensino secundário. As restrições são usadas para configurar o gerador, dando ao utilizador alguma flexibilidade para personalizar as formas das expressões que surgem nos exercícios [34, 38]. Pretendia-se que a variedade dos exercícios gerados contribuísse para melhorar as capacidades de abstração e facilitasse a compreensão de conceitos e procedimentos algorítmicos subjacentes aos mesmos. Mais recentemente, por adaptação do núcleo do sistema AGILMAT, foi desenvolvido um protótipo AgilmatQuiz para gerar automaticamente questões de escolha múltipla para um curso intensivo de Pré-Cálculo, da Faculdade de Ciências da Universidade do Porto (FCUP) [35]. As questões incidem sobre o estudo das funções algébricas básicas lecionadas no ensino secundário e de noções como domínio, contradomínio, imagem recíproca, funções definidas por ramos, além da resolução de equações e inequações no conjunto dos reais. A Figura 1.2 mostra o tipo de exercícios criados.

No âmbito do presente trabalho, considerámos um módulo de geração de exercícios de trigonometria, para extensão dos tópicos suportados pelo protótipo AGILMAT, focando especialmente o cálculo de domínios e a resolução de equações e inequações trigonométricas. Para prova de conceito, trabalhámos sobre uma versão restrita do seu núcleo.

O tipo de exercícios de determinação de domínios e de resolução de equações/inequações que queremos gerar automaticamente com o novo módulo são da forma indicada na Figura 1.3.

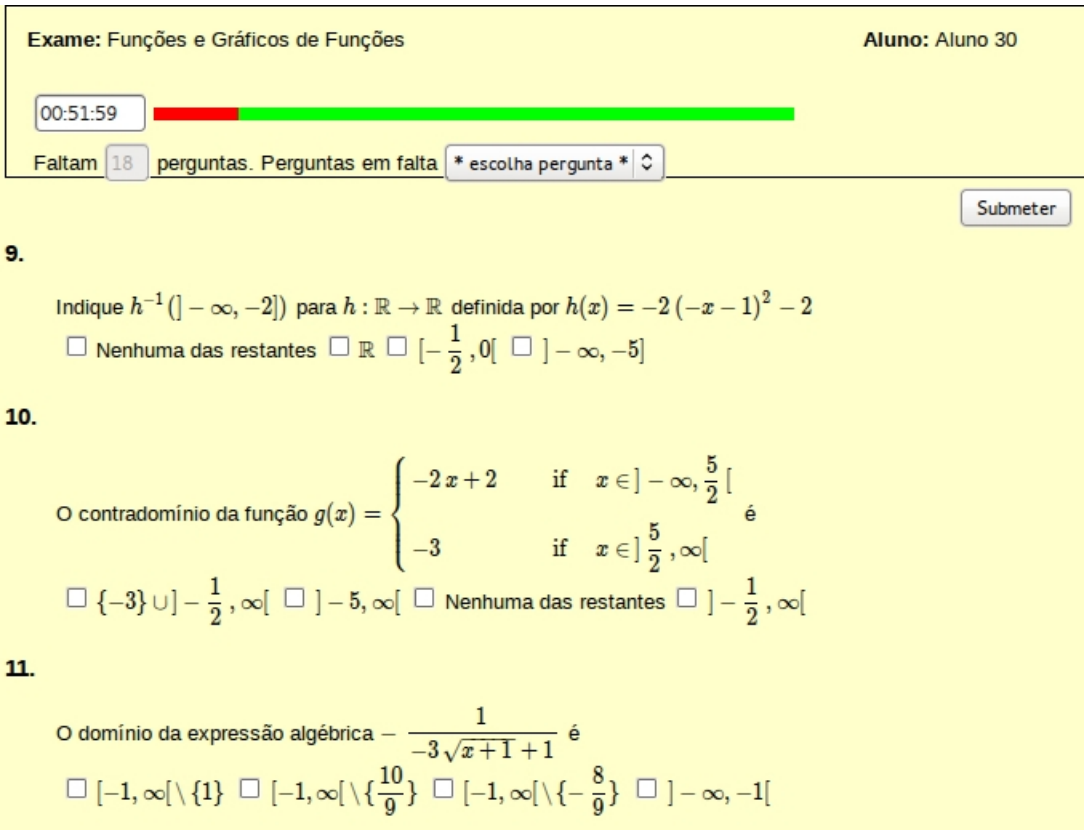


Figura 1.2: Perguntas de escolha múltipla geradas pelo protótipo AgilmatQuiz (fonte [35]).

Encontrar o domínio da expressão f definida em *<domínio>* por $f(x) =$ *<expressão trigonométrica>*

Resolver em *<domínio>* a *<equação/inequação>* *<expressão trigonométrica>* *<operador de comparação>* *<constante>*

Figura 1.3: Tipos de questões de trigonometria a suportar.

Os parâmetros *domínio, expressão trigonométrica, operador de comparação* e *constante* devem ser gerados aleatoriamente, e atender às restrições do utilizador. O parâmetro *equação/inequação* é definido de acordo com o enunciado pretendido.

1.1 Desenvolvimento do trabalho e estrutura da dissertação

Numa primeira fase, a fim de nos familiarizarmos com a abordagem explorada no projeto AGILMAT, começámos por converter alguns módulos do protótipo de SICStus Prolog para SWI-Prolog. A conversão permitiu que o projeto não ficasse dependente de uma licença comercial (ou académica) e se reduzisse a necessidade de bibliotecas de CLP à de `clpfd`. A par, analisámos as formas dos exercícios de trigonometria propostos nos livros de texto das disciplinas de Matemática, no ensino secundário, e procurámos compreender de que modo alguns sistemas existentes os suportam, ou os limitam.

Numa segunda fase, depois de abstrair a forma dos exercícios que pretendemos gerar, desenvolvemos e implementámos uma extensão do gerador de expressões e do resolutor de restrições (equações e inequações) do sistema AGILMAT, para os suportar. Como o protótipo AGILMAT não recorre a sistemas de manipulação algébrica externos, mas integra resolutores específicos, começámos por estudar representações para conjuntos periódicos e as operações de interseção, reunião e diferença sobre tais conjuntos.

Este relatório encontra-se organizado em seis capítulos.

- O Capítulo 1 descreve brevemente o tema e a motivação do projeto de dissertação.
- O Capítulo 2 contém uma descrição sucinta de trabalho relacionado e descreve as ideias centrais da abordagem explorada no projeto AGILMAT.
- O Capítulo 3 foca a extensão definida para o tipo de conjuntos suportados de modo a poder lidar com conjuntos periódicos. Descreve as representações introduzidas e os algoritmos desenvolvidos e implementados para suportar as operações usuais (de união, interseção e diferença) entre conjuntos periódicos e não periódicos, e as suas limitações.
- O Capítulo 4 aborda a geração de expressões e exercícios e a forma como se estendeu a gramática de expressões geradas pelo AGILMAT para incluir funções trigonométricas.
- O Capítulo 5 descreve os métodos implementados para a resolução de exercícios e de que modo condicionam o tipo de exercícios gerados.
- Finalmente, no Capítulo 6, apresentam-se as conclusões do trabalho realizado e apontam-se possíveis extensões.

Capítulo 2

Trabalho Relacionado

Existem inúmeras aplicações informáticas com fins educativos. Algumas são ferramentas interativas que permitem que os alunos pratiquem, analisam as interações dos alunos, fornecem *feedback* sobre as suas ações e ajudam-os a progredir [9]. Neste capítulo efetuamos uma breve revisão de algumas dessas ferramentas e apresentamos os aspetos principais da abordagem explorada no sistema AGILMAT para geração de exercícios.

2.1 Tecnologias Digitais no Ensino

Em Portugal, as calculadoras gráficas ainda constituem a ferramenta mais usada na sala de aula, embora alguns professores utilizem outras aplicações, como GeoGebra [8, 10]. Um inquérito a 44 professores de matemática do Ensino Básico e Secundário, sobre integração de computadores nas práticas pedagógicas, realizado em 2013/2014, concluiu que a Geometria era o tema onde o computador era mais utilizado no ensino de conteúdos matemáticos pelos professores participantes [31].

Entre as tecnologias disponíveis para apoio na sala de aula ou para *e-learning* (ensino eletrónico), encontramos motores de resposta (*Answer Engines* (AE)), sistemas computacionais algébricos (*Computer Algebraic Systems* (CAS)), cursos *online*, incluindo os *Massive Open Online Courses* (MOOCs), e sistemas de apoio à aprendizagem e avaliação (*Assisted Learning and Assessment Systems* (ALAS)). Nesta secção analisamos algumas destas alternativas.

2.1.1 Motores de Resposta e Sistemas Computacionais Algébricos

Os AE assemelham-se aos motores de busca. Enquanto os motores de busca para um dado *input* usam um algoritmo de *ranking* para mostrar as páginas *web* mais relacionadas com tal *input* [6], que

pode ser enviesado pelo perfil do utilizador, os AE interpretam o *input* como uma pergunta e tentam responder de forma concisa e específica dentro da sua "base de conhecimento" [4, 11].

No ensino de Matemática existem várias alternativas de AE mas, neste ponto, destacamos o Wolfram|Alpha [43] que é uma plataforma *web* de fácil acesso. O Wolfram|Alpha tenta responder analiticamente aos problemas usando o Wolfram Mathematica como CAS [15] que, como sistema de computação científica, pode utilizar procedimentos mais avançados para resolver problemas complexos (por exemplo, métodos numéricos para obter soluções aproximadas).

Um dos objetivos da plataforma é ser uma ferramenta de cálculo científico potente e assim conseguir responder ao máximo de perguntas possível [41, 42]. A procura de generalidade e a não verificação de algumas restrições de aplicabilidade podem conduzir a erros. Por exemplo, para a equação $\sin(x) = 2$, atualmente, o Wolfram|Alpha retorna a seguinte resposta:

$$x = 2\pi n + \pi - \sin^{-1}(2), \quad n \in \mathbb{Z}$$
$$x = 2\pi n + \sin^{-1}(2), \quad n \in \mathbb{Z}$$

Isto indica que a aplicação tenta responder sem analisar primeiro a pergunta. Esta falha já tinha sido reportada em 2012 [20, p. 3].

Do ponto de vista pedagógico, os motores de resposta por si só tornam-se num instrumento de exploração, delimitado pelo alcance que têm. Para não se reduzir a "calculadora gráfica" ou apenas "resolutor de exercícios", o Wolfram permite que os alunos resolvam exercícios organizados por temas, dando *feedback* automático [44]. O *blog* da plataforma refere que os exercícios não são recolhidos de uma base de dados mas gerados no momento, mas não explica como [45].

2.1.2 Cursos *Online*

Os cursos *online* surgiram a partir dos anos 1980 [29]. A ideia geral é providenciar um conjunto de conteúdos para aprender e aplicar conhecimentos num dado tema. Atualmente, projetos como Khan Academy [16] ou Udemy [40] atraem tanto jovens estudantes como profissionais pelo grande leque de áreas que abrangem. Estas plataformas são similares entre si, com a exceção de que o Khan Academy não tem fins lucrativos e o Udemy é comercial. Essencialmente, tentam simular a experiência de um curso presencial com aulas gravadas e exercícios relativos à matéria dada em cada aula.

Por muito completos que estes cursos sejam, os exercícios são fixos, o que pode comprometer a consolidação de conhecimentos do aluno.

2.1.3 Sistemas de Apoio à Aprendizagem e Avaliação

Durante os picos mais graves da pandemia por COVID-19, em que foi decretado confinamento, as escolas tiveram de fechar. Enquanto a população recuperava deste fenómeno desastroso, algumas instituições de ensino tiveram duas alternativas para continuar o desenvolvimento dos seus alunos: ou simulavam a sala de aula numa aplicação *web* de vídeo-conferência apenas ou complementavam as aulas *online* com a utilização de sistemas preparados para a gestão da aprendizagem dos alunos quando os professores não estavam presentes fisicamente [27].

Casos como o Moodle [19] ou o ILIAS [12, 13] são exemplos deste tipo de tecnologia, que permitem criar objetivos, submeter documentos e proporcionar momentos de avaliação. Para ter uma experiência mais completa são integrados com sistemas de apoio à aprendizagem e avaliação (ALAS).

O STACK [28, 32] é exemplo de um destes sistemas. Foca-se no suporte aos professores na construção de testes, dando apoio à geração de problemas e das respetivas soluções. Suporta a verificação de respostas com recurso ao sistema de computação algébrica Maxima, como CAS [33].

Em Portugal, mais concretamente na Universidade de Aveiro, existe um projeto de ensino de matemática, o PmatE, onde se produziu o SA^{3}C [14]. Este sistema enquadra-se nos ALAS com a capacidade de gerar problemas de escolha múltipla aleatoriamente, partindo de modelos de exercícios parametrizados [18]. Num dos parâmetros dos modelos de questões é inserida uma expressão. Em [21], é descrita uma aplicação para geração de exercícios para um curso de Cálculo, com expressões construídas a partir de treze modelos parametrizados. O aluno tem uma pontuação associada baseada no número de respostas corretas e o tempo que demorou. O mesmo tipo de abordagem é seguida para construção de exercícios para a competição EQUAmat, que tem participantes desde o 2º ciclo ao secundário.

2.2 AGILMAT

O projeto AGILMAT (POSI/CHS/48565/2002) teve por objetivo a concepção e implementação de uma aplicação *web* para geração automática de exercícios rotineiros de Matemática, facilmente configurável para diversos requisitos [34, 38], como mostra a Figura 2.1, extraída de [35].

De modo a ter interesse pedagógico, devia ficar garantido que os exercícios propostos podiam ser resolvidos pelo computador mas também pelos alunos de um dado nível de ensino. Dever-se-ia ter algum controlo sobre as soluções e aferir e restringir o nível de dificuldade dos exercícios e o seu enquadramento curricular. Para tal, introduziu-se uma nova abordagem para geração das expressões

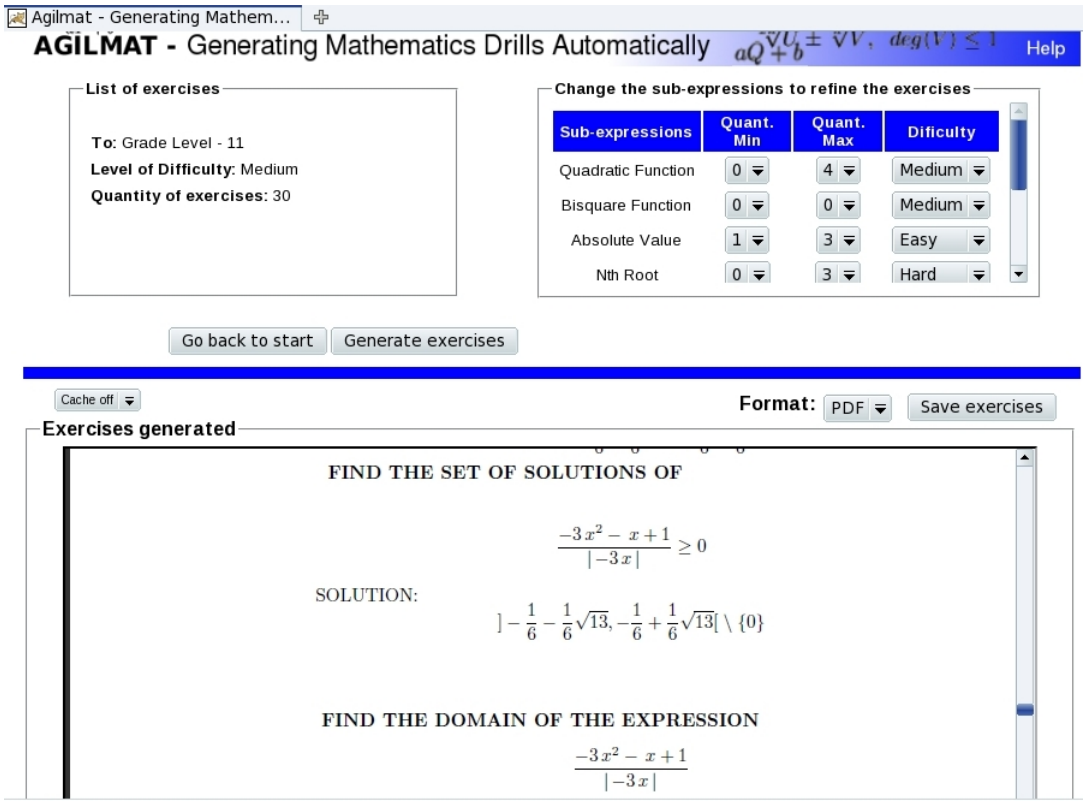


Figura 2.1: Interface do AGILMAT (fonte [35]).

para os exercícios. Em vez de fixar uma forma paramétrica para a expressão incluída no modelo (*template*) que gera a questão [22], partiu-se da análise dos procedimentos algébricos que os alunos conhecem ou aprendem num determinado nível de ensino, para abstrair e restringir as expressões nas perguntas. Esta abordagem é viável quando se consideram exercícios de rotina sobre alguns tópicos e suas soluções numa linha [34] ou soluções passo-a-passo [1, 2]. No sistema AGILMAT, as expressões são especificadas por gramáticas, que podem ser refinadas por restrições de configuração segundo alguns perfis de base e, adicionalmente, por opções do utilizador, como sugere a Fig. 2.1.

O protótipo esteve disponível *online* até 2017, com uma interface *web* de configuração, com divulgação progressiva de parâmetros, dependentes de escolhas do utilizador. O núcleo do sistema é implementado em Prolog, sendo uma aplicação de Programação em Lógica com Restrições (CLP), enquanto que a interface se baseia numa estrutura de assistente *web* (*web wizards framework*) desenvolvido como uma aplicação *web* em Java. A camada frontal, implementada sobre um servidor aplicacional Java, comunica por *streams* (filas) de entrada/saída com um processo Prolog que executa o programa gerador. Esses dois componentes estão fracamente acoplados, como mostra a Figura 2.2, proporcionando um alto grau de independência entre eles. Em modo interativo, com a funcionalidade de *cache* desligada, os exercícios são criados automaticamente em cada interação, por envio de *queries* Prolog ao sistema, que os gerará, segundo os parâmetros indicados.

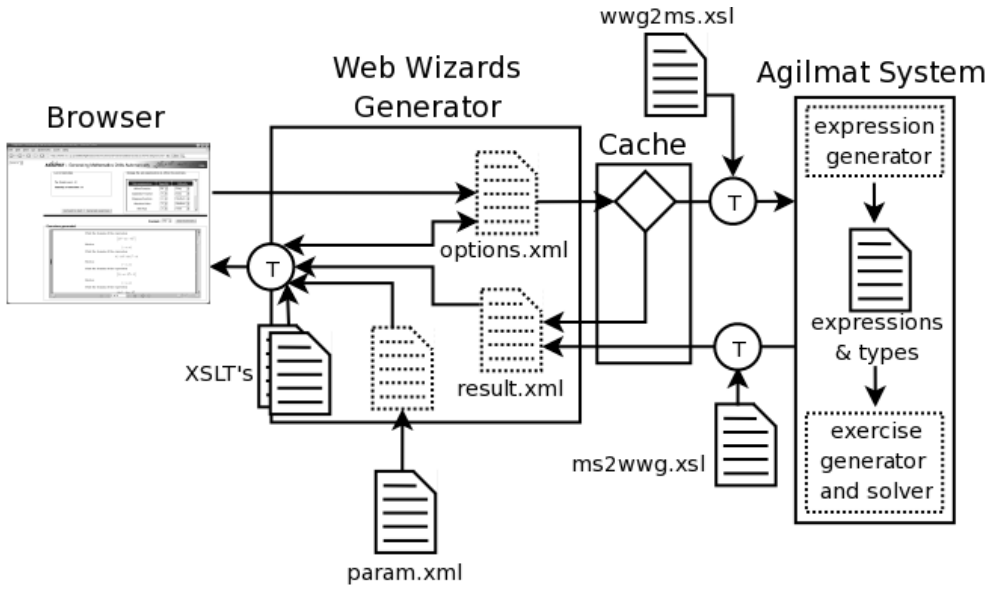


Figura 2.2: Visão geral da arquitetura do sistema AGILMAT (fonte [38]).

O trabalho que realizámos visou a extensão do núcleo, referido na Fig. 2.2 por *Agilmat System*. Teve por objetivo gerar exercícios com funções trigonométricas, com foco no cálculo do domínio de funções e na resolução de restrições da forma $E \trianglelefteq k$, sendo E a expressão gerada, $\trianglelefteq$ um dos operadores relacionais $(=, \neq, <, \leq, >, \geq)$ e k uma constante em $\mathbb{R}$.

O módulo de trigonometria é formado pelo *gerador de expressões* para exercícios e pelo *gerador de exercícios*. Como se esquematiza na Figura 2.3, na *query* ao gerador de expressões, o utilizador especifica restrições sobre os tipos de expressões, nível de dificuldade, grau da expressão (se definido) e o número de instâncias a apresentar para cada tipo de expressão. O gerador produz um ficheiro com

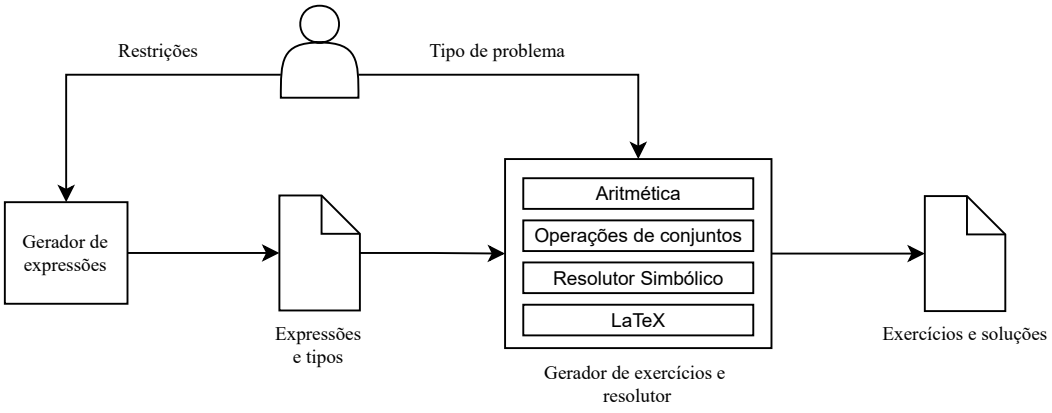


Figura 2.3: Funcionamento do núcleo do sistema AGILMAT (traduzido da apresentação em [38])

as expressões geradas, representadas por termos Prolog. Esse ficheiro pode ser passado ao gerador de exercícios e resolutor para que as expressões sejam usadas nos enunciados das questões.

O utilizador escolhe o tipo de problema que pretende. Caso prefira cálculo de domínios para funções

de variável real, as expressões são suficientes, embora seja possível ter variantes do exercício com restrição do domínio a `Dom` $\subset \mathbb{R}$. O predicado `domain_expr(Expr,X,Dom,Domf)`, implementado no resolutor para calcular o domínio `Domf` da expressão `Expr` na variável `X`, recebe `Dom` como argumento o qual, por defeito, pode corresponder a $\mathbb{R}$. Caso o utilizador pretenda exercícios sobre equações/inequações, terá de indicar os operadores de comparação que podem ser usados e um número $k \in \mathbb{R}$. Para cada candidato a exercício, definido por uma expressão `Expr`, um operador relacional `Op`, escolhido aleatoriamente, e uma constante `K`, o gerador de exercícios começa por calcular o domínio `Dom` de `Expr`, chamando o predicado `domain_expr/4`. De seguida, chama o predicado `solving(Expr,Op,K,X,Dom,Sols)` para obter as solucões `Sols` da equação/inequação nesse domínio, sendo `X` a variável que ocorre na expressão. Se o resolutor conseguir obter a solução (isto é, se `solving` não falhar), o exercício e a solução são passados ao módulo de escrita em LaTeX. Senão, o exercício é descartado e passa à próxima expressão do ficheiro. O gerador termina quando não encontra mais expressões no ficheiro dado. No fim, é retornado um ficheiro com os exercícios e as suas soluções em LaTeX. Um esquema de geração de exercícios na opção de cálculo de restrições é apresentado na Figura 2.4.

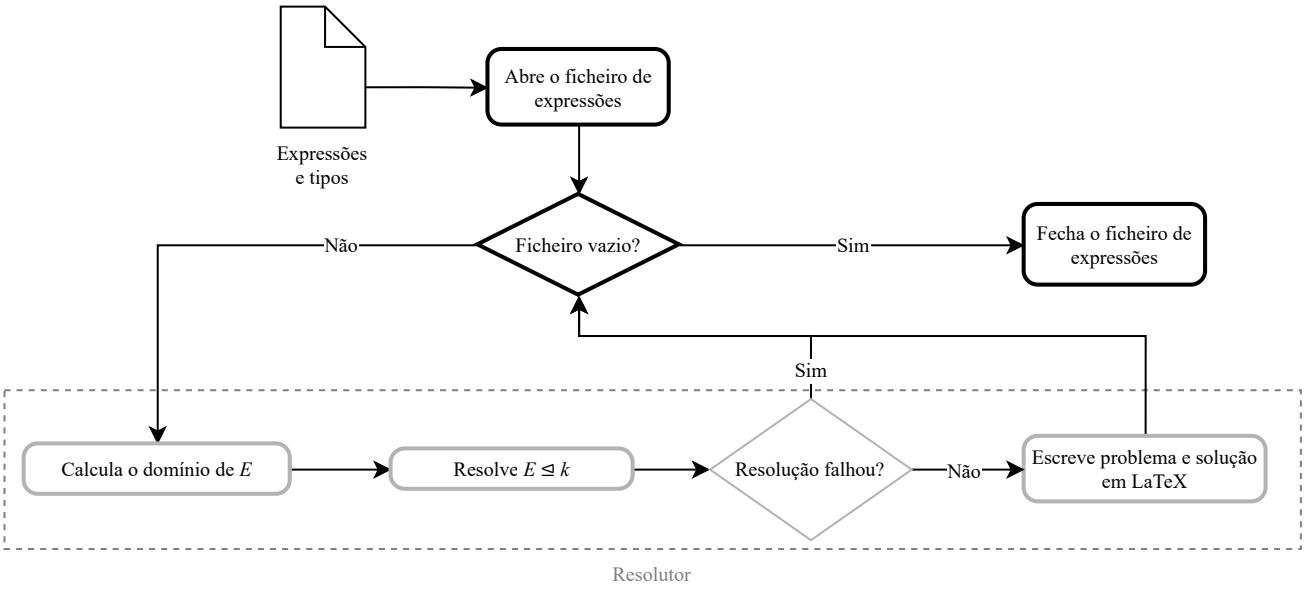


Figura 2.4: Passos do gerador de exercícios na opção de resolução de restrições, onde E representa uma expressão do ficheiro de expressões.

Capítulo 3

Suporte de conjuntos periódicos

Quando as equações/inequações incluem apenas funções algébricas (i.e, polinomiais, racionais, valor absoluto, radicais e potência), o conjunto de soluções em $\mathbb{R}$ é união finita de intervalos e/ou de pontos isolados, ou conjunto vazio (isto é, $\emptyset$) ou $\mathbb{R}$. Por esta razão, as representações e operações sobre conjuntos definidas anteriormente para o sistema AGILMAT assumiam tais formas.

Exemplo 3.1 *Por exemplo, o domínio da função*

$$f(x) = \frac{3\sqrt{x+3}-5}{|x^3-4x^2|}$$

corresponde a $x \geq -3 \wedge x \notin \{-2, 0, 2\}$ *que se equivale a*

$$[-3, -2[\cup]-2, 0[\cup]0, 2[\cup]2, +\infty[$$

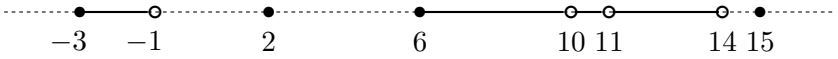
e o conjunto das soluções de $f(x) \geq 0$*, para* $x \in \mathbb{R}$*, é*

$$\left[-\frac{2}{9}, 0\right[\cup]0, 2[\cup]2, +\infty[$$

o qual define o conjunto dos pontos x *do domínio de* f *que satisfazem* $x \geq -\frac{2}{9}$.

Definição 3.1 (forma standard [37]) *Um conjunto de números reais está em* forma standard *se é* $\emptyset$ *ou um intervalo ou um subconjunto finito de pontos isolados (i.e., um conjunto discreto finito) ou a união de uma sequência finita* $S_1, \ldots, S_n$ *de intervalos ou conjuntos discretos finitos, disjuntos dois a dois, tais que para* $1 \leq i < n$*, tem-se* S_i *ou* S_{i+1} *não é um conjunto discreto,* $\sup(S_i) \leq \inf(S_{i+1})$*, e se* $\sup(S_i) = \inf(S_{i+1})$ *então* $\sup(S_i) \notin S_i$ *e* $\inf(S_{i+1}) \notin S_{i+1}$*. O ínfimo e o supremo de cada conjunto podem ser* $-\infty$ *e* $+\infty$*, respetivamente. Para salientar que se trata de conjuntos disjuntos dois a dois, escrevemos* $S_1 \uplus S_2 \uplus \cdots \uplus S_n$*, usando* $\uplus$ *em vez de* $\cup$*. Um conjunto que pode ser representado na* forma standard *designa-se por* conjunto restritivo *ou "constraining set".*

Exemplo 3.2 *O conjunto* $([-3,-1[\uplus \{2,15\} \uplus [6,11[\uplus]11,14[) \setminus \{10\}$ *de números reais é um* "constraining set"*. Na forma standard é dado por* $[-3,-1[\uplus \{2\} \uplus [6,10[\uplus]10,11[\uplus]11,14[\uplus \{15\}$*, que corresponde à leitura da sua representação na reta real, da esquerda para a direita.*



Para que o resolutor do sistema AGILMAT pudesse determinar as soluções de equações/inequações trigonométricas em $\mathbb{R}$, foi necessário introduzir uma representação para conjuntos periódicos e suportar as operações usuais, ainda que com algumas limitações. Neste capítulo focamos essa extensão, que tem por base trabalho iniciado em [36]. Começamos por apresentar algumas noções.

3.1 Conjuntos periódicos em forma standard

Definição 3.2 (Conjunto periódico) *Um subconjunto* S *de* $\mathbb{R}$ *é* **periódico** *com período* $p \in \mathbb{R}^+$ *se e somente se* $x + pk \in S$*, quaisquer que sejam* $x \in S$ *e* $k \in \mathbb{Z}$*. Geometricamente,* S *é periódico com período* p *se* S *é* **invariante por translações por** p **(ou múltiplos de** p**).** *Para qualquer conjunto* $A \subseteq \mathbb{R}$ *e número positivo* $p \in \mathbb{R}$*, o conjunto* $A \oplus p\mathbb{Z}$*, definido como* $\{a + pk \mid a \in A,\ k \in \mathbb{Z}\}$*, é um conjunto periódico com período* p*.*

De acordo com esta definição, $\mathbb{R}$ e $\emptyset$ são conjuntos periódicos para qualquer período $p \in \mathbb{R}^+$. Dado um conjunto periódico S, o *período fundamental* de S, se existir, é o menor $p \in \mathbb{R}^+$ tal que $x + p \in S$, para todo $x \in S$.

Estamos interessados em conjuntos periódicos $A \oplus p\mathbb{Z}$, em que $A \neq \emptyset$ é um *"constraining set"* limitado (superiormente e inferiormente), $p \in \mathbb{R}^+$ é o período, e $\mathbb{R} \neq A \oplus p\mathbb{Z}$.

Exemplo 3.3 *As soluções de* $0 < \sin(x) \leq \frac{1}{2}$*, para* $x \in \mathbb{R}$*, são dadas por*

$$0 + 2\pi k < x \leq \frac{\pi}{6} + 2\pi k \quad \vee \quad \frac{5\pi}{6} + 2\pi k \leq x < \pi + 2\pi k, \quad \text{com } k \in \mathbb{Z}.$$

O conjunto-solução é um conjunto periódico, que definimos sucintamente por $(\ ]0, \frac{\pi}{6}] \cup [\frac{5\pi}{6}, \pi[\ ) \oplus 2\pi\mathbb{Z}$*, sendo* 2π *o período.*

Definição 3.3 (Conjunto periódico em forma standard [36]) *Seja* A *um* "constraining set"*, tal que* $\emptyset \neq A \neq \mathbb{R}$ *e* A *é limitado. Seja* p *um número real positivo, tal que o conjunto periódico induzido por* A *e* p*, denotado por* $A \oplus p\mathbb{Z}$ *não é* $\mathbb{R}$*. Então,* $A \oplus p\mathbb{Z}$ *está em* forma standard *se* p *é período,* A *está na forma standard e* $(A \oplus p\mathbb{Z}) \cap [0, p[= A$*.*

Exemplo 3.4 (cont. Exemplo 3.3) *Na forma standard, o conjunto soluções de* $0 < \sin(x) \leq 1/2$*, para* $x \in \mathbb{R}$*, é dado por* $(\ ]0, \frac{\pi}{6}]\ \uplus [\frac{5\pi}{6}, \pi[\ ) \oplus 2\pi\mathbb{Z}$*, sendo* 2π *o período.*

3.1.1 Representação simbólica

Nesta exposição, e na implementação em Prolog, consideramos que $A \oplus p\mathbb{Z}$ é representado por um termo $\texttt{periodic}(A, p)$, e assumimos que:

- $A = S_1 \uplus S_2 \uplus \cdots \uplus S_n$ é dado por uma lista (de termos), em que cada S_i é um intervalo ou um ponto isolado (desagregamos conjuntos discretos);
- um intervalo é representado como $\texttt{interval}(L(x), R(y))$, com $L, R \in \{\texttt{a}, \texttt{f}\}$, em que `a` é abreviatura de *aberto* e `f` é abreviatura de *fechado*;
- se $S_i = \texttt{interval}(L(x), R(y))$ então definimos $L(S_i) = L(x)$ e $R(S_i) = R(y)$;
- um ponto isolado x é representado por $\texttt{i}(x)$;
- nas operações sobre conjuntos não-periódicos, $\sup(\cdot)$ e $\inf(\cdot)$ têm como parâmetro um intervalo ou ponto independente e retornam o seu supremo e ínfimo, respetivamente;
- por abuso de notação, escrevemos $\sup(\texttt{i}(x)) = \inf(\texttt{i}(x)) = x$, para referirmos o supremo de $\{x\}$ e o ínfimo de $\{x\}$;
- para facilitar a descrição das operações de união e interseção, se $X(x)$ e $Y(x)$ forem extremos de intervalos, relativos ao mesmo ponto x, definimos operações $X(x) \wedge Y(x)$ e $X(x) \vee Y(x)$ por:

$$X(x) \wedge Y(x) = \begin{cases} \texttt{f}(x) & \text{se } X = Y = \texttt{f} \\ \texttt{a}(x) & \text{caso contrário} \end{cases}$$

$$X(x) \vee Y(x) = \begin{cases} \texttt{a}(x) & \text{se } X = Y = \texttt{a} \\ \texttt{f}(x) & \text{caso contrário} \end{cases}$$

Nas próximas secções, apresentamos as operações sobre conjuntos implementadas no resolutor.

3.2 Operação de expansão

Para calcular, por exemplo, a interseção de um conjunto periódico $S = A \oplus p\mathbb{Z}$ com um intervalo $I = [a, b]$, com $a, b \in \mathbb{R}$, pode ser necessário expandir a representação, isto é, determinar um conjunto auxiliar $S' = A \oplus p\,\mathcal{K}$, com $\mathcal{K} \subset \mathbb{Z}$ finito, tal que $\sup(S') \geq \sup(I) = b$ e $\inf(S') \leq \inf(I) = a$, devendo $\mathcal{K}$ ser mínimo.

Exemplo 3.5 *Para* $S = [0,3] \oplus 5\mathbb{Z}$ *e* $I = [-8,9]$*, o conjunto auxiliar seria*

$$S' = [-10,-7] \uplus [-5,-2] \uplus [0,3] \uplus [5,8] \uplus [10,13],$$

correspondendo a $A \oplus p\mathcal{K}$*, com* $\mathcal{K} = \{-2,-1,0,1,2\}$*.*

Tal expansão é obtida por adição ou subtração sucessiva do período p, um certo número de vezes, k, de modo que $a \leq x + kp \leq b$, o que implica que

$$\inf(A) + kp \leq a \quad \Leftrightarrow \quad k \leq \frac{a - \inf(A)}{p}$$
$$\sup(A) + kp \geq b \quad \Leftrightarrow \quad k \geq \frac{b - \sup(A)}{p}$$

e, consequentemente, para ter $\mathcal{K}$ mínimo, os valores de $k \in \mathcal{K}$ satisfazem

$$\left\lfloor \frac{a - \inf(A)}{p} \right\rfloor \leq k \leq \left\lceil \frac{b - \sup(A)}{p} \right\rceil$$

No Exemplo 3.5, teríamos $-2 = \left\lfloor -\frac{8}{5} \right\rfloor \leq k \leq \left\lceil \frac{6}{5} \right\rceil = 2$, e resultaria o conjunto indicado.

O algoritmo de expansão é simples, baseando-se num processo iterativo, em que se vai criando S' por ordem crescente, de modo que, dado $S = A \oplus p\mathbb{Z}$ na forma standard, o conjunto S' determinado também estará representado na forma standard.

3.3 Operação de interseção

Sendo X e Y conjuntos dados na forma standard, iremos estender a operação anteriormente definida para interseção de conjuntos não periódicos de forma a suportar a interseção de periódicos e a interseção de um periódico com um não periódico que é um "constraining set" limitado (superiormente e inferiormente). Para tal, definimos métodos específicos $\cap_{np}$, $\cap_p$ e $\cap_{pnp}$ para não periódicos (np), periódicos (p) e um periódico e um intervalo (pnp)

$$X \cap Y = \begin{cases} \emptyset & \text{se } X = \emptyset \text{ ou } Y = \emptyset \\ Y & \text{se } X = \mathbb{R} \\ X & \text{se } Y = \mathbb{R} \\ X \cap_p Y & \text{se } X \text{ e } Y \text{ são conjuntos periódicos} \\ X \cap_{np} Y & \text{se } X \text{ e } Y \text{ são conjuntos não-periódicos} \\ X \cap_{pnp} Y & \text{se } X \text{ for conjunto periódico e } Y \text{ for um "constraining set" limitado} \end{cases}$$

Nos restantes casos, a operação não será suportada. Para um termo $X \texttt{ cap } Y$, que representa internamente $X \cap Y$, o predicado retorna $X \texttt{ cap } Y$ intocado. Caso contrário, retorna a representação de $X \cap Y$ na forma standard.

A seguir descrevemos as operações $\cap_{np}$, $\cap_{p}$ e $\cap_{pnp}$, com algum detalhe, que traduz a análise de casos subjacente à sua implementação no sistema.

3.3.1 Interseção entre conjuntos não-periódicos

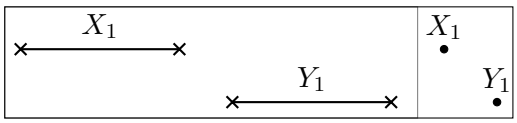
Sejam X e Y conjuntos dados na forma standard, sendo $X = \emptyset$ ou $X = X_1$ ou $X = \uplus_{k=1}^{n} X_k$ e $Y = \emptyset$ ou $Y = Y_1$ ou $Y = \uplus_{k=1}^{m} Y_k$ respetivamente, com $n > 1$ e $m > 1$, onde X_k é $\texttt{interval}(L(x), R(y))$ ou $\{\texttt{i}(x)\}$. Para evitar sobrecarregar a descrição, sem perda de generalidade, iremos focar o caso $\sup(X_1) \leq \sup(Y_1)$. Se ocorrer o inverso, $\sup(Y_1) < \sup(X_1)$, basta trocar Y com X.

Com algum abuso de notação, escrevemos $\texttt{i}(x)$ em vez de $\{\texttt{i}(x)\}$ e assumimos que $\uplus_{k=a}^{b} X_k = \emptyset$, se $b < a$, e $\uplus_{k=a}^{b} X_k = X_a$, se $a = b$, e do mesmo modo para Y.

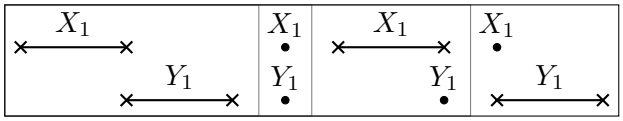
O conjunto $Z = X \cap_{np} Y$ na forma *standard* é assim definido:

1. Caso $X = \emptyset$ ou $Y = \emptyset$: $\quad Z = \emptyset$

2. Caso $\sup(X_1) < \inf(Y_1)$: $\quad Z = (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$



3. Caso $\sup(X_1) = \inf(Y_1) \ = x$:

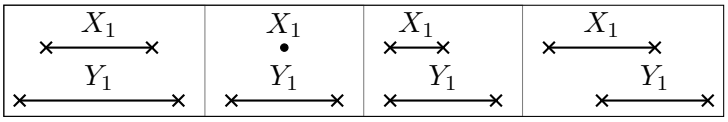


   3.1. $x \in X_1$ e $x \in Y_1$: $\quad Z = \texttt{i}(x) \ \uplus \ (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$

   3.2. $x \notin X_1$ ou $x \notin Y_1$: $\quad Z = (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$

4. Caso $\inf(Y_1) < \sup(X_1)$ e $Y_1 = \texttt{interval}(L(Y_1), R(Y_1))$:

   4.1. $\sup(X_1) < \sup(Y_1)$:

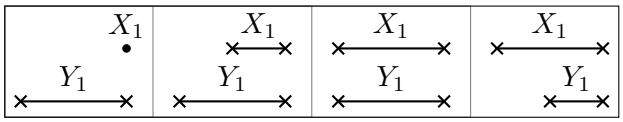


      4.1.1. $\inf(X_1) > \inf(Y_1)$: $\quad Z = X_1 \ \uplus \ (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$

      4.1.2. $\inf(X_1) = \inf(Y_1)$: $\quad Z = \texttt{interval}(L(X_1) \wedge L(Y_1), R(X_1)) \ \uplus \ (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$

4.1.3. $\inf(X_1) < \inf(Y_1)$: $Z = \texttt{interval}(L(Y_1), R(X_1)) \uplus (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=1}^{m} Y_k)$

4.2. $\sup(X_1) = \sup(Y_1)$:



4.2.1. $X_1 = \texttt{i}(x)$:

4.2.1.1. $x \in Y_1$: $Z = \texttt{i}(x) \uplus (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=2}^{m} Y_k)$

4.2.1.2. $x \notin Y_1$: $Z = (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=2}^{m} Y_k)$

4.2.2. $\inf(X_1) > \inf(Y_1)$:

$Z = \texttt{interval}(L(X_1), R(X_1) \wedge R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=2}^{m} Y_k)$

4.2.3. $\inf(X_1) = \inf(Y_1)$:

$Z = \texttt{interval}(L(X_1) \wedge L(Y_1), R(X_1) \wedge R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=2}^{m} Y_k)$

4.2.4. $\inf(X_1) < \inf(Y_1)$:

$Z = \texttt{interval}(L(Y_1), R(X_1) \wedge R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cap_{np} \uplus_{k=2}^{m} Y_k)$

3.3.2 Interseção entre conjuntos periódicos

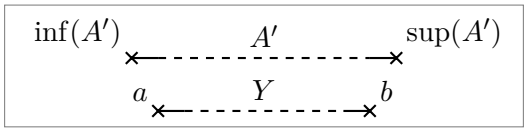
Sejam X e Y conjuntos periódicos dados por $X = A \oplus p\,\mathbb{Z}$ e $Y = B \oplus q\,\mathbb{Z}$. Estes são, respetivamente, representados simbolicamente por termos $\texttt{periodic}(A, p)$ e $\texttt{periodic}(B, q)$, em que A e B estão na forma *standard*, $X \cap [0, p[= A$ e $Y \cap [0, q[= B$. Definimos $X \cap_p Y$ por

$$X \cap_p Y = (A' \cap_{np} B') \oplus \texttt{mmc}(p, q)\,\mathbb{Z}$$

em que A' e B' são as expansões de A e B em $[0, \texttt{mmc}(p, q)[$ e $\texttt{mmc}(p, q)$ é o mínimo múltiplo comum de p e q. Caso $p = q$, tal corresponde a $(A \cap_{np} B) \oplus p\,\mathbb{Z}$.

3.3.3 Interseção entre conjuntos periódicos e não periódicos

Sejam $X = A \oplus p\,\mathbb{Z}$ e Y um "constraining set" limitado. Pode ser necessário expandir A, para obter $A' = A \oplus p\,\mathcal{K}$, tal que $A' \supseteq [a, b]$, com $a = \inf(Y)$ e $b = \sup(Y)$. Portanto, o primeiro objetivo, esquematizado na imagem, é expandir A, se necessário.



Depois da expansão, $X \cap_{pnp} Y = A' \cap_{np} Y$.

Notamos que $A' \cap_{np} Y$ pode ser $\emptyset$.

3.4 Operação de união

Tal como para a interseção, restringimos o suporte a apenas algumas formas. Dados X e Y em forma standard, definimos $X \cup Y$ por

$$X \cup Y = \begin{cases} Y & \text{se } X = \emptyset \\ X & \text{se } Y = \emptyset \\ \mathbb{R} & \text{se } X = \mathbb{R} \text{ ou } Y = \mathbb{R} \\ X \cup_{np} Y & \text{se } X \text{ e } Y \text{ são conjuntos não periódicos} \\ X \cup_{p} Y & \text{se } X \text{ e } Y \text{ são conjuntos periódicos} \end{cases}$$

onde $\cup_{np}$ e $\cup_p$ designam união entre conjuntos não periódicos e entre conjuntos periódicos.

Nos restantes casos, a operação não será suportada: para um termo $X \,\texttt{cup}\, Y$, que internamente representa $X \cup Y$, retorna $X \,\texttt{cup}\, Y$.

3.4.1 União de conjuntos não-periódicos

Sejam X e Y conjuntos não periódicos em forma standard. Enquanto que para a interseção de conjuntos não periódicos, as regras definidas produzem $Z = X \cap_{np} Y$ na forma standard, para a união, será primeiramente construída uma forma intermédia, que depois será reduzida à forma standard.

As regras que apresentamos a seguir, para a **Fase I** da união, preservam para o resultado Z a condição $\sup(Z_i) \leq \inf(Z_{i+1})$ mas, quando $\sup(Z_i) = \inf(Z_{i+1})$, podemos ter $\sup(Z_i) \in Z_i$ ou $\inf(Z_{i+1}) \in Z_{i+1}$, o que não pode acontecer na forma *standard*. Para reduzir Z à forma standard, aplicamos o método descrito na **Fase II** mais abaixo.

As regras para obter $Z = X \cup_{np} Y$, na primeira fase, que apresentamos a seguir, assumem $\sup(X_1) \leq \sup(Y_1)$. Se ocorrer o inverso, basta trocar Y com X.

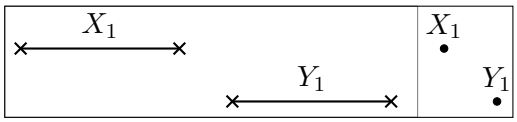
Fase I

1. Caso $X = \emptyset$ ou $Y = \emptyset$:

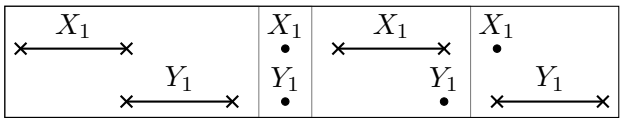
   1.1. Caso $X = \emptyset$: $\quad Z = Y$

   1.2. Caso $Y = \emptyset$: $\quad Z = X$

2. Caso $\sup(X_1) < \inf(Y_1)$: $\quad Z = X_1 \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=1}^{m} Y_k)$



3. Caso $\sup(X_1) = \inf(Y_1) = x$:



3.1. $x \in X_1$ ou $x \in Y_1$:

3.1.1. $X_1 = \mathtt{i}(x)$ e $Y_1 = \mathtt{i}(x)$: $\quad Z = \mathtt{i}(x) \uplus (\uplus_{k=2}^{n} X_k \cup \uplus_{k=2}^{m} Y_k)$

3.1.2. $X_1 = \mathtt{interval}(L(X_1), R(x))$ e $Y_1 = \mathtt{i}(x)$:

$$Z = \mathtt{interval}(L(X_1), \mathtt{f}(x)) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=2}^{m} Y_k)$$

3.1.3. $X_1 = \mathtt{i}(x)$ e $Y_1 = \mathtt{interval}(L(x), R(Y_1))$:

$$Z = (\uplus_{k=2}^{n} X_k \cup_{np} (Y_1' \uplus \uplus_{k=2}^{m} Y_k)), \text{com } Y_1' = \mathtt{interval}(\mathtt{f}(x), R(Y_1))$$

3.1.4. $X_1 = \mathtt{interval}(L(X_1), R(x))$ e $Y_1 = \mathtt{interval}(L(x), R(Y_1))$:

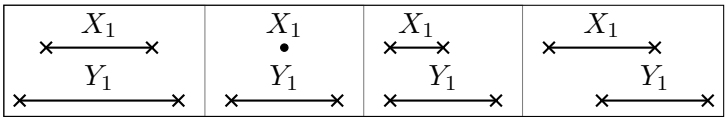
$$Z = (\uplus_{k=2}^{n} X_k \cup_{np} (Y_1' \uplus \uplus_{k=2}^{m} Y_k)), \text{ com } Y_1' = \mathtt{interval}(L(X_1), R(Y_1))$$

3.2. $x \notin X_1$ e $x \notin Y_1$:

$$Z = X_1 \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=1}^{m} Y_k)$$

4. Caso $\sup(X_1) > \inf(Y_1)$ e $Y_1 = \mathtt{interval}(L(Y_1), R(Y_1))$:

4.1. $\sup(X_1) < \sup(Y_1)$:



4.1.1. $X_1 = \mathtt{i}(x)$ ou $\inf(X_1) > \inf(Y_1)$:

$$Z = \mathtt{interval}(L(Y_1), \mathtt{f}(\inf(X_1))) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} (Y_1' \uplus \uplus_{k=2}^{m} Y_k)),$$

com $Y_1' = \mathtt{interval}(\mathtt{a}(\inf(X_1)), R(Y_1))$

4.1.2. $\inf(X_1) = \inf(Y_1)$

$$Z = \mathtt{interval}(L(X_1) \vee L(Y_1), \mathtt{f}(\sup(X_1))) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} (Y_1' \uplus \uplus_{k=2}^{m} Y_k)),$$

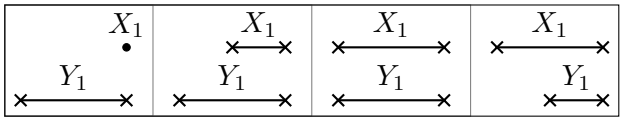
com $Y_1' = \mathtt{interval}(\mathtt{a}(\sup(X_1)), R(Y_1))$

4.1.3. $\inf(X_1) < \inf(Y_1)$:

$$Z = \texttt{interval}(L(X_1), \texttt{f}(\sup(X_1))) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} (Y_1' \uplus \uplus_{k=2}^{m} Y_k)),$$

com $Y_1' = \texttt{interval}(\texttt{a}(\sup(X_1)), R(Y_1))$

4.2. $\sup(X_1) = \sup(Y_1)$:



4.2.1. $X_1 = \texttt{i}(\texttt{x})$:

$$Z = \texttt{interval}(L(Y_1), \texttt{f}(x)) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=2}^{m} Y_k)$$

4.2.2. $\inf(X_1) > \inf(Y_1)$:

$$Z = \texttt{interval}(L(Y_1), R(X_1) \vee R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=2}^{m} Y_k)$$

4.2.3. $\inf(X_1) = \inf(Y_1)$:

$$Z = \texttt{interval}(L(X_1) \vee L(Y_1), R(X_1) \vee R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=2}^{m} Y_k)$$

4.2.4. $\inf(X_1) < \inf(Y_1)$:

$$Z = \texttt{interval}(L(X_1), R(X_1) \vee R(Y_1)) \uplus (\uplus_{k=2}^{n} X_k \cup_{np} \uplus_{k=2}^{m} Y_k)$$

Fase II - redução à forma *standard*

Seja S o conjunto resultante da união. Então, $S = \emptyset$ ou $S = S_1$ ou $S = \uplus_{k=1}^{n} S_k$, para algum $n > 1$. Para todo k, o conjunto S_k é um intervalo ou um ponto isolado e S satisfaz

- $\sup(S_k) \leq \inf(S_{k+1})$, para todo $k < n$
- $S_k \neq S_j$, para todo (j, k) com $j \neq k$

Caso $S = \emptyset$ ou $S = S_1$, o valor de $\texttt{reduz}(S) = S$. Nos restantes casos, $Z = \texttt{reduz}(\uplus_{k=1}^{n} S_k)$ é assim definido:

1. $\sup(S_1) < \inf(S_2)$: $Z = S_1 \uplus \texttt{reduz}(\uplus_{k=2}^{n} S_k))$
2. $\sup(S_1) = \inf(S_2) = x$:

   2.1. $S_1 = \texttt{i}(x)$ e $S_2 = \texttt{interval}(L(S_2), R(S_2))$: $Z = \texttt{reduz}(S')$ para

$$S' = \texttt{interval}(\texttt{f}(x), R(S_2)) \uplus \uplus_{k=3}^{n} S_k$$

2.2. $S_2 = \texttt{i}(x)$ e $S_1 = \texttt{interval}(L(S_1), R(S_1))$: $Z = \texttt{reduz}(S')$ para

$$S' = \texttt{interval}(L(S_1), \texttt{f}(x)) \uplus \uplus_{k=3}^{n} S_k$$

2.3. $S_1 = \texttt{interval}(L(S_1), R(S_1))$ e $S_2 = \texttt{interval}(L(S_2), R(S_2))$:

2.3.1. $R(S_1) \vee L(S_2) = \texttt{f}(x)$: $Z = \texttt{reduz}(S')$ com

$$S' = \texttt{interval}((L(S_1), R(S_2)) \uplus \uplus_{k=3}^{n} S_k$$

2.3.2. $R(S_1) \vee L(S_2) = \texttt{a}(x)$: $Z = S_1 \uplus \texttt{reduz}(\uplus_{k=2}^{n} S_k)$

3.4.2 União entre conjuntos periódicos

Sejam X e Y conjuntos periódicos dados por $X = A \oplus p\,\mathbb{Z}$ e $Y = B \oplus q\,\mathbb{Z}$. Estes são, respetivamente, representados simbolicamente por termos $\texttt{periodic}(A, p)$ e $\texttt{periodic}(B, q)$, em que A e B estão na forma *standard*, $X \cap [0, p[= A$ e $Y \cap [0, q[= B$. Como para a interseção de periódicos, definimos $X \cup_p Y$ por

$$X \cup_p Y = (A' \cup_{np} B') \oplus \texttt{mmc}(p, q)\,\mathbb{Z}$$

em que A' e B' são as expansões de A e B em $[0, \texttt{mmc}(p, q)[$ e $\texttt{mmc}(p, q)$ é o mínimo múltiplo comum de p e q. Caso $p = q$, tal corresponde a $(A \cup_{np} B) \oplus p\,\mathbb{Z}$.

3.5 Operação de diferença

Sendo X e Y conjuntos na forma standard, a diferença $X \setminus Y$ é suportada também com algumas restrições, para

$$X \setminus Y = \begin{cases} \emptyset & \text{se } X = \emptyset \text{ ou } Y = \mathbb{R} \\ X & \text{se } Y = \emptyset \\ X \setminus_{np} Y & \text{se } X \text{ e } Y \text{ são conjuntos não periódicos} \\ X \setminus_{npp} Y & \text{se } X \text{ for não periódico e } Y \text{ for periódico} \end{cases}$$

onde $\setminus_{np}$ e $\setminus_{npp}$ implementam as regras específicas para não periódicos e para um par (não periódico, periódico). Nos restantes casos, a operação não será suportada: para um termo $X\,\texttt{setminus}\,Y$, que internamente representaria $X \setminus Y$, retorna $X\,\texttt{setminus}\,Y$.

3.5.1 Diferença entre conjuntos não-periódicos

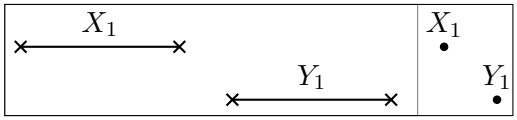
Como para a interseção, dados conjuntos não periódicos X e Y, representados na forma standard, diferença $Z = X \setminus Y$, pode ser obtida em forma standard por aplicação das regras seguintes.

1. Caso $X_1 = \emptyset$ ou $Y_1 = \emptyset$:

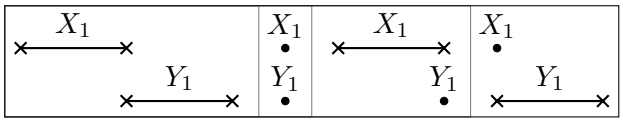
   1.1. Caso $X = \emptyset$: $Z = \emptyset$

   1.2. Caso $Y = \emptyset$: $Z = X$

2. Caso $\sup(X_1) < \inf(Y_1)$: $Z = X_1 \uplus (\uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k)$



3. Caso $\sup(X_1) = \inf(Y_1) = x$:



   3.1. $X_1 = \mathtt{i}(x)$:

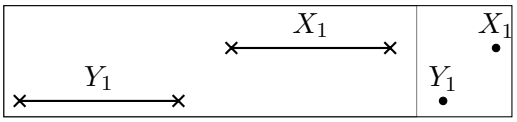
$$Z = \begin{cases} \uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k & \text{se } x \in Y_1 \\ X_1 \uplus (\uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k) & \text{caso contrário} \end{cases}$$

   3.2. $X_1 = \mathtt{interval}(L(X_1), R(x))$:

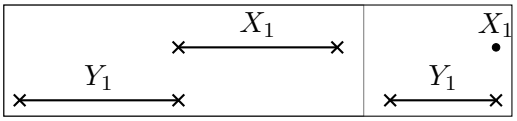
$$Z = \begin{cases} \mathtt{interval}(L(X_1), \mathtt{a}(x)) \uplus (\uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k) & \text{se } x \in Y_1 \\ X_1 \uplus (\uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k) & \text{caso contrário} \end{cases}$$

4. $\sup(X_1) > \inf(Y_1)$:

   4.1. $\sup(Y_1) < \inf(X_1)$: $Z = \uplus_{k=1}^{n} X_k \setminus \uplus_{k=2}^{m} Y_k$



   4.2. $\sup(Y_1) = \inf(X_1) = x$:



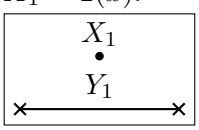
      4.2.1. $x \notin Y_1$: $Z = \uplus_{k=1}^{n} X_k \setminus \uplus_{k=2}^{m} Y_k$

      4.2.2. $x \in Y_1$:

$$Z = \begin{cases} \uplus_{k=2}^{n} X_k \setminus \uplus_{k=2}^{m} Y_k & \text{se } X_1 = i(x) \\ (\mathtt{interval}(\mathtt{a}(x), R(X_1)) \uplus_{k=2}^{n} X_k) \setminus \uplus_{k=2}^{m} Y_k & \text{se } X_1 = \mathtt{interval}((L(X_1), R(X_1)) \end{cases}$$

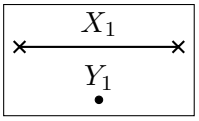
   4.3. $\sup(Y_1) > \inf(X_1) = x$:

      4.3.1. $X_1 = \mathtt{i}(x)$: $Z = \uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k$



4.3.2. $X_1 = \texttt{interval}(L(X_1), R(X_1))$:

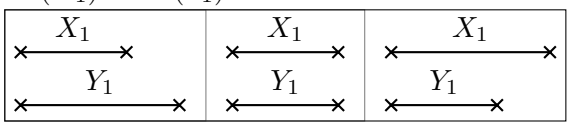
4.3.2.1. $Y_1 = \texttt{i}(y)$:



$$Z = \texttt{interval}(L(X_1), \texttt{a}(y)) \uplus ((\texttt{interval}(\texttt{a}(y), R(X_1)) \uplus \uplus_{k=2}^{n} X_k) \setminus \uplus_{k=2}^{m} Y_k)$$

4.3.2.2. $Y_1 = \texttt{interval}(L(Y_1), R(Y_1))$:

4.3.2.2.a. $\inf(X_1) = \inf(Y_1) = x$:



4.3.2.2.a.1. $x \notin X_1$:

4.3.2.2.a.1.1. $\sup(X_1) < \sup(Y_1)$:

$$Z = \uplus_{k=2}^{n} X_k \setminus \uplus_{k=1}^{m} Y_k$$

4.3.2.2.a.1.2. $\sup(X_1) = \sup(Y_1) = y$:

$$Z = \begin{cases} \uplus_{k=2}^{n} X_k \setminus \uplus_{k=2}^{m} Y_k & \text{se } (y \notin X_1) \vee (y \in X_1 \wedge y \in Y_1) \\ \texttt{i}(y) \uplus (\uplus_{k=2}^{n} X_k \setminus \uplus_{k=2}^{m} Y_k) & \text{se } y \in X_1 \wedge y \notin Y_1 \end{cases}$$

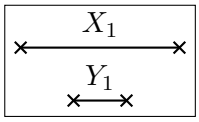
4.3.2.2.a.1.3. $\sup(X_1) > \sup(Y_1) = y$:

$$Z = (\texttt{interval}(\texttt{a}(y), R(X_1)) \uplus \uplus_{k=2}^{m} X_k) \setminus \uplus_{k=2}^{m} Y_k$$

4.3.2.2.a.2. $x \in X_1$:

$$Z = \begin{cases} \texttt{i}(x) \uplus ((\texttt{interval}(\texttt{a}(x), R(X_1)) \uplus \uplus_{k=2}^{n} X_k) \setminus \uplus_{k=1}^{m} Y_k) & \text{se } x \notin Y_1 \\ (\texttt{interval}(\texttt{a}(x), R(X_1)) \uplus \uplus_{k=2}^{n} X_k) \setminus \uplus_{k=1}^{m} Y_k & \text{se } x \in Y_1 \end{cases}$$

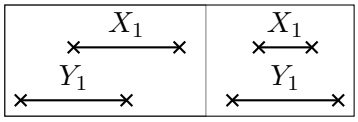
4.3.2.2.b. $\inf(X_1) < \inf(Y_1) = y$:



$$Z = \begin{cases} \texttt{interval}(L(X_1), \texttt{a}(y)) \uplus (X' \setminus \uplus_{k=1}^{m} Y_k) & \text{se } y \in Y_1 \\ \texttt{interval}(L(X_1), \texttt{f}(y)) \uplus (X' \setminus \uplus_{k=1}^{m} Y_k) & \text{se } y \notin Y_1, \end{cases}$$

onde $X' = (\texttt{interval}(\texttt{a}(y), R(X_1)) \uplus \uplus_{k=2}^{n} X_k)$

4.3.2.2.c. $x = \inf(X_1) > \inf(Y_1)$:



$$Z = (\texttt{interval}(\texttt{a}(x), R(X_1)) \uplus \uplus_{k=2}^{n} X_k) \setminus (\texttt{interval}(\texttt{a}(x), R(Y_1)) \uplus \uplus_{k=1}^{m} Y_k)$$

3.5.2 Diferença entre um conjunto não periódico e um conjunto periódico

Sejam X um conjunto não periódico e $Y = B \oplus p\,\mathbb{Z}$ um periódico, ambos na forma standard. A operação de diferença implementada não cobre todos os casos, mas tem os necessários para poder suportar os exercícios pretendidos.

- Se X for vazio, $X \setminus_{npp} Y = X = \emptyset$.
- Caso $X = \mathbb{R}$, a diferença $\mathbb{R} \setminus_{npp} Y$ é o complementar de Y em $\mathbb{R}$. Podemos determinar o complementar de B em $[0, p[$, que denotamos por $B^C = [0, p[\setminus_{np} B$, e ter como resultado $\mathbb{R} \setminus_{npp} Y = B^C \oplus p\mathbb{Z}$.
- Se X for um conjunto limitado, determinamos $Y^C = \mathbb{R} \setminus_{npp} Y$, e calculamos $X \cap_{npp} Y^C$.
- Se X for um conjunto não limitado superiormente ou inferiormente (por exemplo, quando X corresponde a $\mathbb{R}^+$ ou a $\mathbb{R}^-$), mas $X \neq \mathbb{R}$, é retornada a expressão recebida sem alterações.

Capítulo 4

Geração de Questões de Trigonometria

Este capítulo aborda a geração de expressões para os exercícios e a forma como se estendeu a gramática de expressões geradas pelo AGILMAT para incluir funções trigonométricas. Descreve também como se geram exercícios sobre resolução de equações/inequações trigonométricas, que usam tais expressões.

4.1 Tipos e instâncias de expressões

Segundo [34, 38], a característica que distingue o sistema AGILMAT das ferramentas de *e-learning* que utilizam bases de dados de exercícios ou modelos fixos (*templates*), em que a variação se resume à alteração de constantes, é o facto de o seu gerador de expressões poder potencialmente produzir expressões arbitrariamente complexas.

As expressões são caracterizadas por uma gramática [34], que descreve um conjunto não trivial de expressões cujos zeros e domínios podem ser calculados exatamente pelos procedimentos algébricos que os alunos devem dominar (até ao 12º ano).

Por meio de restrições, impostas pelo perfil selecionado e por opções do utilizador, as formas podem ser refinadas. O gerador de expressões é um programa Prolog, que usa módulos de programação com restrições, para permitir lidar com restrições sobre variáveis inteiras, com domínios finitos, que controlam os tipos gerados.

Cada expressão gerada segue um padrão, que é denotado por um *tipo*. Por exemplo, as expressões da forma $\sqrt[N]{(Ax+B)^M}$, com parâmetros A, B, N, M, são representadas por termos `rad(pow(pol(x,[A,B]),M,N))` e consideradas do *tipo* `rad(N) o pow(M) o p1 o x`. A composição é denotada por `o` e `p1` representa uma função polinomial de grau 1, `rad(N)` e `pow(M)`

representam raíz de expoente N e a potência de expoente M. Em particular, para

$$\sqrt[3]{|-4(3x-4)^2-2|+(x-3)^2}$$

a representação interna do *tipo* e da *expressão* corresponde a

```
rad(3)o(abs o p1 o pow(2)o p1 o x + pow(2)o p1 o x)
rad(abs(pol(pow(pol(x,[3,-4]),2),[-4,-2])) + pow(pol(x,[1,-3]),2),3)
```

Os parâmetros de configuração restringem esses tipos. As constantes que surgem na expressão (coeficientes e expoentes de radicais e potências) também podem ser restringidas. O gerador produz tipos e uma (ou mais) expressões de cada tipo. Desta forma, é possível produzir várias expressões com exatamente o mesmo tipo mas coeficientes distintos.

Para a resolução de exercícios, o sistema AGILMAT não recorre a resolutores externos. Integra um resolutor específico, implementado em Prolog e que, naturalmente, é mais limitado do que um sistema de computação algébrica (CAS).

4.2 Extensão para incluir exercícios de trigonometria

Seguindo a abordagem do projeto AGILMAT, para abstrair a forma das expressões mais comuns nas questões de trigonometria, analisámos livros de exercícios do 11º e 12º anos do ensino português. O estudo da trigonometria no ensino secundário é mais focado nas funções seno, cosseno e tangente, embora as funções inversas (arco-seno, arco-cosseno e arco-tangente) também sejam lecionadas, com recurso à calculadora gráfica. As funções cotangente, secante e cossecante não são mencionadas nas metas curriculares da disciplina de Matemática A [3].

Como referimos no Capítulo 2, no desenvolvimento do sistema AGILMAT separou-se, em certa medida, a geração de expressões da geração dos exercícios em que as expressões são incluídas. Contudo, para tratar funções definidas por ramos, o resolutor tem de ser usado, para garantir algum ajuste nos ramos da função produzida [35].

Do mesmo modo, na criação de exercícios sobre equações e inequações com funções trigonométricas, é necessário interligar as duas componentes, para garantir que o exercício é adequado ao nível de ensino. Por exemplo, se a questão fosse "Resolver em $\mathbb{R}$ a inequação $\text{sen}(x) \leq 3/4$", os alunos não teriam forma de apresentar a solução exata sem utilizar a função arco-seno. A solução S seria

$$S = \left[-\pi - \text{arcsen}\left(\frac{3}{4}\right), \text{arcsen}\left(\frac{3}{4}\right)\right] \oplus 2\pi\mathbb{Z}.$$

ou seja,

$$-\pi - \operatorname{arcsen}\left(\frac{3}{4}\right) + 2\pi k \ \leq x \ \leq \operatorname{arcsen}\left(\frac{3}{4}\right) + 2\pi k, \ \text{ com } k \in \mathbb{Z}.$$

No entanto, sendo $\operatorname{sen}(x)$ é uma expressão válida e, se a questão fosse "Resolver em $\mathbb{R}$ a equação $\operatorname{sen}(x) = \sqrt{3}/2$", o aluno deveria estar apto a responder, pois espera-se que conheça os valores de $\operatorname{sen}(\alpha)$ para os ângulos fundamentais, isto é, para $\alpha \in \{0, \pi/6, \pi/4, \pi/3, \pi/2, \pi\}$. A diversidade dos padrões de expressões que ocorrem nos exercícios de trigonometria é muito menor do que nos anteriormente gerados envolvendo apenas funções algébricas simples, e há que atender a restrições como as que acabámos de referir.

Importa notar também que o tratamento de certas matérias ou tipos de problemas de Matemática pode ser computacionalmente intratável ou mesmo indecidível. Não é por isso de estranhar que muitos sistemas existentes na web se foquem em níveis básicos de ensino ou limitem drasticamente a diversidade do que produzem.

4.2.1 Extensão da gramática

Para garantir que são geradas funções no âmbito do programa curricular, as funções são representadas de forma composta $f(p(x))$, em que f é uma função trigonométrica básica (seno, cosseno, tangente) e a função p é uma função polinomial de 1º grau. Posto isto, identificámos duas categorias:

- **trig** que denota expressões simples, da forma $f(p(x))$, em que f é uma função trigonométrica básica (seno, cosseno, tangente) e $p(x)$ uma polinomial de grau 1;
- **sumt** que denota expressões do tipo $Af(x) + Bg(x)$, onde A e B são constantes, $A, B \in \mathbb{R}$ e f e g são da categoria **trig**.

Como **sumt** é demasiado geral, definimos antes duas subcategorias, mais restritas:

sumtrig para expressões em que $A, B \in \{-1, 1\}$, e que, no caso de a função básica em f ser uma tangente também terá que ser uma tangente em g, mas se for um seno ou um cosseno, também o pode ser em g, arbitrariamente.

sumtrig$_{\mathbf{AB}}$ para expressões $A\sin(p(x)) + B\cos(p(x))$ que se reduzem a uma forma $C\cos(p(x) \pm \alpha)$, sendo C e α determinadas por A e B. Neste caso, as duas funções polinomiais terão de ser iguais. Para simplificar uma expressão do tipo **sumtrig$_{\mathbf{AB}}$**, usamos a regra da adição

$$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta)$$

Exemplo 4.1 *Se* $A = 1$ *e* $B = \sqrt{3}$*, podemos simplificar*

$$\sin(x) + \sqrt{3}\cos(x) = \sqrt{1^2 + (\sqrt{3})^2}\left(\frac{1}{\sqrt{1^2 + (\sqrt{3})^2}}\sin(x) + \frac{\sqrt{3}}{\sqrt{1^2 + (\sqrt{3})^2}}\cos(x)\right)$$
$$= 2\left(\frac{1}{2}\sin(x) + \frac{\sqrt{3}}{2}\cos(x)\right) = 2\left(\sin\left(\frac{\pi}{6}\right)\sin(x) + \cos\left(\frac{\pi}{6}\right)\cos(x)\right) = 2\cos(x - \frac{\pi}{6}).$$

Visto que trabalharemos somente com os valores β do seno e cosseno para os ângulos fundamentais, podemos assumir que

$$\beta = \frac{A}{\sqrt{A^2 + B^2}} \in \left\{\frac{1}{2}, \frac{\sqrt{3}}{2}, \frac{\sqrt{2}}{2}\right\}$$

e por isso A e B terão de se relacionar por

$$\frac{A}{\sqrt{A^2 + B^2}} = \beta \Leftrightarrow \frac{A^2}{A^2 + B^2} = \beta^2 \Leftrightarrow A^2 = \beta^2 A^2 + \beta^2 B^2$$
$$\Leftrightarrow (1 - \beta^2)A^2 = \beta^2 B^2 \Leftrightarrow A^2 = \frac{\beta^2 B^2}{1 - \beta^2}$$
$$\Leftrightarrow \left(A = \frac{\beta B}{\sqrt{1 - \beta^2}} \wedge B > 0\right) \vee \left(A = -\frac{\beta B}{\sqrt{1 - \beta^2}} \wedge B < 0\right)$$

Consequentemente, A terá de ser função de B e β e satisfazer a relação (4.1).

$$A = \begin{cases} B\frac{\sqrt{3}}{3} & \text{se } B > 0 \wedge \beta = \frac{1}{2} \\ -B\frac{\sqrt{3}}{3} & \text{se } B < 0 \wedge \beta = \frac{1}{2} \\ \sqrt{3}B & \text{se } B > 0 \wedge \beta = \frac{\sqrt{3}}{2} \\ -\sqrt{3}B & \text{se } B < 0 \wedge \beta = \frac{\sqrt{3}}{2} \\ B & \text{se } B > 0 \wedge \beta = \frac{\sqrt{2}}{2} \\ -B & \text{se } B < 0 \wedge \beta = \frac{\sqrt{2}}{2} \end{cases} \tag{4.1}$$

A Figura 4.1 mostra a extensão proposta para a gramática definida em [34], contemplando estas categorias e a variante **vtrig**, que permite a composição com valor absoluto, potência e raíz. Embora não esteja escrito na gramática, as constantes A e B nas expressões da categoria $\mathbf{sumtrig_{AB}}$ terão de seguir as restrições indicadas em (4.1). A Figura 4.2 apresenta esquematicamente as categorias que constituem *vtrig*.

Como na abordagem anterior, continuaremos a manter representações para *tipos* e para *expressões*, que são instâncias de um *tipo*. Para a implementação da extensão em Prolog, foram definidas as representações que constam da Figura 4.3.

$$
\begin{aligned}
function &\longrightarrow (\texttt{k}*)^{?} prodfact \mid (\texttt{k}*)^{?} divexpr \\
prodfact &\longrightarrow factor \mid prodexpr \\
divexpr &\longrightarrow prodfact/prodfact \mid \texttt{k}/prodfact \mid prodfact/\texttt{k} \\
&\longrightarrow \texttt{pow}(divexpr, N) \mid \texttt{rad}(divexpr, N) \mid \texttt{abs}(prodexpr) \\
factor &\longrightarrow sumexpr \mid vxip \mid basic \mid vtrig \\
vtrig &\longrightarrow trig \mid sumtrig \mid sumtrig_{AB} \mid abs(vtrig) \mid pow(vtrig, N) \mid rad(vtrig, N) \\
sumtrig &\longrightarrow a * \texttt{sin}(f\,pol_1(\texttt{x}))) + b * \texttt{sin}(f\,pol_1(\texttt{x}))) \\
&\longrightarrow a * \texttt{sin}(f\,pol_1(\texttt{x}))) + b * \texttt{cos}(f\,pol_1(\texttt{x}))) \\
&\longrightarrow a * \texttt{cos}(f\,pol_1(\texttt{x}))) + b * \texttt{cos}(f\,pol_1(\texttt{x}))) \\
&\longrightarrow a * \texttt{tan}(f\,pol_1(\texttt{x}))) + b * \texttt{tan}(f\,pol_1(\texttt{x}))),\ a, b = \{1, -1\} \\
sumtrig_{AB} &\longrightarrow A * \texttt{sin}(\texttt{pol}(x, [a, b])) + B * \texttt{cos}(\texttt{pol}(x, [a, b])), a \neq 0 \text{ e } A, B, b \in \mathbb{R} \\
trig &\longrightarrow \texttt{sin}(f\,pol_1(\texttt{x}))) \mid \texttt{cos}(f\,pol_1(\texttt{x})) \mid \texttt{tan}(f\,pol_1(\texttt{x}))) \\
f\,pol_1(T) &\longrightarrow \texttt{pol}(T, [a, b]), a \in \mathbb{R} \setminus 0, b \in \mathbb{R} \\
\texttt{x} &\longrightarrow \text{variável} \\
\texttt{k} &\longrightarrow \text{constante}
\end{aligned}
$$

Figura 4.1: Excerto da gramática atualizada

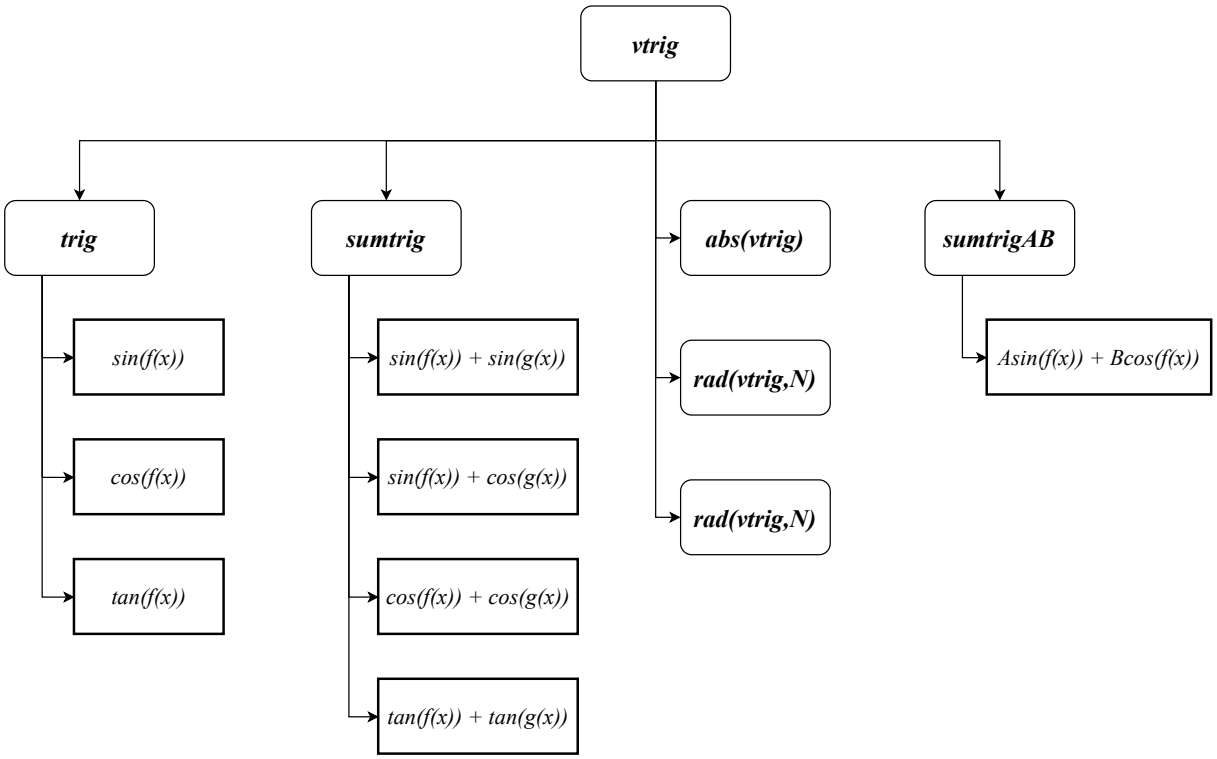


Figura 4.2: Categoria *vtrig* em árvore, sendo f e g são funções polinomiais de grau um.

Tipo	Representação simbólica	Expressão
trig `o p1 o x`	*trig*(`pol(x,`$[a,b]$`)`)	$f(ax+b)$
`sumtrig(`F,G`) o x`	`sumtrig(`$P,Q,a,b,[F,G]$`)`	$aF(P)+bG(Q)$
`sumtrigAB(`F',G'`) o x`	`sumtrigAB(`$P,a',b',[F',G']$`)`	$a'F'(P)+b'G'(P)$
`abs o` *trig* `o p1 o x`	`abs(`*trig*`(pol(x,`$[a,b]$`)))`	$\lvert f(ax+b)\rvert$
`pow(`N`) o` *trig* `o p1 o x`	`pow(`*trig*`(pol(x,`$[a,b]$`)),N)`	$f(ax+b)^N$
`rad(`N`) o` *trig* `o p1 o x`	`rad(`*trig*`(pol(x,`$[a,b]$`)),N)`	$\sqrt[N]{f(ax+b)}$

Figura 4.3: Representações para tipos e para expressões implementadas na extensão, sendo $f \in \{\sin, \cos, \tan\}$, *trig* $\in \{$`sin`, `cos`, `tan`$\}$, F e G combinação de `sin` e `cos` ou ambos `tan`, F' e G' combinação sem repetição de `sin` com `cos`, P e Q do tipo `p1`, $a, b \in \{1, -1\}$, $a', b' \in \mathbb{R}$ e $N \in \mathbb{N}$.

Deste modo, o gerador pode criar expressões como

$$\frac{4}{\left|\sin\left(-x+\frac{\pi}{2}\right)\right|}$$

que é uma instância do tipo

```
k/abs o sin o p1 o x
```

e representada pelo termo

```
4/abs(sin(pol(x,[rat(-1,1),rat(1,2)*symbc(pi)]))).
```

Nesta representação interna, um termo `rat`(P,Q) identifica um número racional, com numerador P e denominador Q. O termo `symbc(pi)` é a representação simbólica para π. Na adaptação do gerador, considerámos que os coeficientes das expressões do tipo `p1` que ocorrem como argumento de funções trigonométricas podem ser racionais e envolver π. Na versão anterior do gerador, as expressões geradas tinham coeficientes inteiros, razão pela qual o valor `4` surge como instância da expressão de tipo `k`.

4.3 Construção de Exercícios

Nesta secção descrevemos alguns detalhes da implementação, relativos à geração das expressões e construção de exercícios.

4.3.1 Geração de Expressões

É útil poder estimar e condicionar o nível de dificuldade dos exercícios. Se a questão for "Determinar os zeros das funções $f(x) = \sin(x)$ e $g(x) = 3x^2 + 2$", a resolução do problema para $f(x)$ requer conhecimentos mais avançados do que para $g(x)$. Na implementação do gerador do AGILMAT, as regras da gramática são traduzidas por predicados da forma

```
category(Type,Degree,Rate,Ts,Ops),
```

sendo `function(Type,Degree,Rate,Ts,Ops)` o principal. Os parâmetros `Degree`, `Rate`, `Ts`, e `Ops` são utilizados para restringir o tipo `Type` gerado. Dessa forma é possível controlar o nível de dificuldade e suportar as restrições impostas pelo utilizador.

`Degree`: permite controlar o grau da expressão gerada. Interessa sobretudo para a geração de funções polinomiais (por exemplo, de grau 1 ou 2) mas na geração de expressões com funções trigonométricas a noção de grau perde o sentido.

`Ops`: é o número de operações (composições, somas, produtos e quocientes) na expressão. Na geração, `Ops` pode ser usada para filtrar algumas simetrias (na presença de somas, por exemplo) [34] e para impor restrições sobre o número total de operadores (para controlar a complexidade da expressão).

`Rate`: define o nível da dificuldade do tipo criado, como soma dos níveis de dificuldade atribuídos a funções primitivas (`p1`, `p2`, `abs`, `rad(_)`, `pow(_)`, ...) e a subexpressões particulares (soma de radicais, quocientes, e produtos, ...). Na Figura 4.4 podemos ver um exemplo do valor atribuído a cada tipo elementar. Seria de esperar que o tipo `sumtrig(_,_)` tivesse associado um `Rate` maior, por ser soma de dois *trig*. Porém o valor total contabilizará os *rate* dos argumentos também. Assim, embora `sumtrig(_,_)` tenha apenas *rate* de 1, como envolve duas funções trigonométricas, ficará com um *rate* total de 11 = 5+5+1. Para o tipo `sumtrigAB(_,_)`, adotamos a mesma abordagem por ser considerado um tipo de exercício de maior dificuldade, embora o resultado possa ser diferente.

Com os valores tabelados, o nível 11 inclui expressões como $a(|cx^2 + dx + e|) + b$, que são do tipo `p1 o abs o p2 o x`, o que permite ter uma ideia do nível total de dificuldade atribuído a uma expressão `sumtrig(_,_)`.

Contadores de tipos `Ts`: lista de variáveis (de domínio finito) que permite limitar o número de ocorrências de cada tipo constituinte da expressão. Para configuração do gerador, impõem-se restrições sobre as categorias, que são especificadas por termos da forma `T-[`n_{min}, n_{max}`]`,

Tipo	Rate
`p1`	2
`p2`	4
`abs`	5
`rad(_)`	4
`power(_)`	4
`divstype`	3
`sin`	5
`cos`	5
`tan`	5
`sumtrig(_,_)`	1
`sumtrigAB(_,_)`	3

Figura 4.4: Exemplo de níveis de dificuldade atribuídos a alguns tipos.

onde n_{min} e n_{max} definem o número mínimo e máximo de ocorrências do tipo `T` como componente. Para gerar expressões com uma ocorrência da função seno e nenhuma das restantes trigonométricas, a lista de configuração para `function` poderia ser

```
[abs-[0,1],power(_)-[0,1],rad(_)-[0,1]
   sin-[1,1],cos-[0,0],tan-[0,0],
   sumtrig(_,_)-[0,0], sumtrigAB(_,_)-[0,0], ...].
```

forçando a entrada de um `sin`, e permitindo ter uma ocorrência de `abs`, `power(_)` e `rad(_)`. A lista aqui representada não está completa (deveria conter quase todos os tipos usados pelo gerador).

A Figura 4.5 mostra um excerto da implementação do predicado `vtrigtype`, referente à categoria *vtrig* da gramática proposta, apresentada anteriormente na Fig. 4.1.

A implementação mantém uma estrutura semelhante à usada no AGILMAT [34, 35]. As três claúsulas definem três alternativas para criação do tipo: ser uma função trigonométrica simples (`trig`) ou uma soma de funções trigonométricas (`sumtrig` ou `sumtrigAB`) ou a composição de uma função do tipo `npftype` (que engloba `abs`, `rad` e `pow`) com uma do tipo `vtrigtype`.

Na primeira claúsula, `sum/3` restringe a soma dos contadores de tipos a 2, e `incr_restr/2` incrementa o contador para `T`, contabilizando o tipo T e, a seguir, o tipo `p1`. Na segunda claúsula, representamos os casos das somas de funções trigonométricas, onde os predicados `sumtrig_restrs/3`

```
vtrigtype(T o p1 o x,_G,Rate,Ts,2):-
   trig_type(T), rate(T,Rate),
   sum(Ts,#=,2), incr_restr(T,Ts), incr_restr(p1,Ts).
vtrigtype(T o x,_G,Rate,Ts,5):-
   (sumtrig_restrs(T,Ts,Rate); sumtrigAB_restrs(T,Ts,Rate)).
vtrigtype(T o Tc,_G,Rate,Ts,Ops) :-
   npftype(T),
   rate_restr(T,Rate,[RateC]),
   types_restr(T,Ts,[TsC]),
   ops_restr(Ops,1,[OpsC]),
   degree(T,_Gt),
   c_trigtype(T,Tc), vtrigtype(Tc,_Gc,RateC,TsC,OpsC).
```

Figura 4.5: Implementação da geração da categoria *vtrig*.

e `sumtrigAB_restrs/3` verificam a combinatória entre as diferentes funções subjacentes e atualizam os contadores de tipos e o `Rate`. Contrariamente às duas primeiras, a terceira claúsula não é terminal. Na chamada recursiva de `vtrigtype`, o tipo `Tc` está condicionado pela escolha de `T`, sendo tal determinado pela chamada de `c_trigtype/2`.

No programa, a geração de uma expressão de um dado tipo é efetuada pelo predicado `expr/3`. A Fig. 4.6 mostra um excerto da implementação que obtém uma instância de `sumtrigAB(_,_)`.

```
expr(sumtrigAB(Tf,Tg) o x, X, Er):-
   Er = sumtrigAB(pol(X,[Pa,Pb]),A,B,[Tf,Tg]),
   sumtrigAB_type(sumtrigAB(Tf,Tg)),
   trig_pol_coefs(Pa,Pb),
   sumtrigAB_coefs(A,B).
```

Figura 4.6: Excerto do predicado `expr/3` para o tipo `sumtrig(_,_)`.

O tipo `sumtrigAB(_,_)` é sempre composto com a identidade, pelo que, nesta claúsula, não há uma chamada recursiva de `expr`. O predicado `sumtrigAB_type/1` verifica se o tipo recebido é válido. Depois, `trig_pol_coefs/2` constroi duas funções polinomiais de primeiro grau, que seguem as regras da categoria $fpol_1(T)$ da gramática, e `sumtrigAB_coefs` gera os coeficientes A, B de acordo com as regras em (4.1).

4.3.2 De expressões para exercícios

No âmbito desta dissertação, considerámos a resolução de equações e inequações trigonométricas, trabalhando sobre uma versão restrita do núcleo do protótipo AGILMAT.

Para criar o exercício, o gerador recebe um ficheiro com pares tipo-expressão, que definem a expressão a incluir no exercício. O nosso objetivo para o módulo de trigonometria é resolver restrições em que as expressões são dos tipos enumerados na Figura 4.3. Na resolução de uma restrição, pode ser necessário caraterizar o domínio da expressão, pelo que o cálculo de domínios constitui um dos problemas tratados.

Na interface mais usual, a geração de exercícios produz um ficheiro LaTeX que contém pares exercício e solução. Só são apresentados os exercícios que o resolutor consegue resolver. Se o resolutor falhar, o gerador de exercícios descarta esse par tipo-expressão e avança para o próximo par.

4.3.2.1 Exercícios que envolvem restrições

As restrições que resolvemos seguem a forma $E \trianglelefteq k$, onde E é uma expressão gerada, $k \in \mathbb{R}$ e o operador $\trianglelefteq \in \{<, >, \leq, \geq, =, \neq\}$. O operador $\trianglelefteq$ é definido aleatoriamente dentro de uma lista de operadores indicada pelo utilizador. A definição do k não é arbitrária, pois tem forçosamente de conduzir a valores conhecidos para cada função trigonométrica. Para isso, analisamos o tipo da expressão e, dependendo desse tipo, é gerado um k dentro de um conjunto restrito de valores. Por exemplo, para o tipo de expressões que tenham `sin` ou `cos`,

$$k_{\texttt{sin}}, k_{\texttt{cos}} \in \left\{-1, -\frac{\sqrt{3}}{2}, -\frac{\sqrt{2}}{2}, -\frac{1}{2}, 0, \frac{1}{2}, \frac{\sqrt{2}}{2}, \frac{\sqrt{3}}{2}, 1\right\},$$

para o tipo `tan`,

$$k_{\texttt{tan}} \in \left\{-\sqrt{3}, -1, -\frac{\sqrt{3}}{3}, 0, \frac{\sqrt{3}}{3}, 1, \sqrt{3}\right\}.$$

Mesmo assim, estes conjuntos não são suficientes para representar a generalidade dos casos que tratamos. Alguns tipos de expressões exigem mais ou menos limitações sobre k. Por exemplo, se tivermos uma inequação em que a expressão é do tipo `sumtrig(sin,sin)`, corremos o risco de gerar

$$\sin(x) + \sin\left(x + \frac{\pi}{7}\right) > \frac{1}{2},$$

e não teríamos forma de resolver. Porém, seria aceitável se fosse

$$\sin(x) + \sin\left(x + \frac{\pi}{7}\right) > 0.$$

A tabela na Figura 4.7 mostra as condições impostas sobre k em cada caso. São apresentados apenas excertos das expressões porque o surgimento de um tipo pertencente à categoria *vtrig* identifica

o método de decisão sobre a constante k. Se a expressão for qualquer outro caso não previsto pela Figura 4.7, k é escolhido aleatoriamente dentro do conjunto $\mathbb{U}$ que é definido a partir da constante u que é identificada pelo utilizador. O conjunto $\mathbb{U}$ é definido pelas regras em 4.2.

$$\mathbb{U} = \begin{cases} \{0\}, \text{ se } u = 0 \\ \{-1, 1\}, \text{ se } u \in \{1\} \\ \{-u, u+1, \ldots, u\}, \text{ caso contrário.} \end{cases} \tag{4.2}$$

Tipo	k
trig `o x`	k_{trig}
`sumtrig(_,_)`	0
`sumtrigAB(f,g)`	$k_{sin}\left(\sqrt{A^2 + B^2}\right)$
`power(`N`) o` *trig*	$(k_{trig})^N$
`rad(`N`) o trig`	$\sqrt[N]{k_{trig}^{0+}}$, se N é par
`rad(`N`) o trig`	$\sqrt[N]{k_{trig}}$, se N é ímpar
`rad(`N`) o trig`	$\sqrt[N]{k_{trig}}$, se N é ímpar
`abs o` *trig*	k_{trig}

Figura 4.7: Lista de pares tipo-constante, sendo *trig* $\in \{\texttt{sin}, \texttt{cos}, \texttt{tan}\}$ e A e B os coeficientes de `f` e `g`, respetivamente.

Habitualmente, a questão requer soluções que pertençam a intervalo (limitado) que é um subconjunto de $\mathbb{R}$. Para tal, depois de ter a restrição criada, é necessário escolher esse intervalo.

4.3.2.2 Determinação de um intervalo para o enunciado do exercício

Para que a questão tenha interesse pedagógico, a escolha do intervalo $I = \texttt{interval}(L(a), R(b))$ a apresentar para procura de soluções deve ser criteriosa. Antes de iniciarmos a explicação do método que usamos para o obter, introduzimos algumas noções:

- Chamamos *elemento constituinte* a cada intervalo ou ponto isolado de um conjunto. Por exemplo, o conjunto $[-2, 0[\cup [5, 10] \cup \{12\}$ tem três elementos constituintes;
- O conjunto $P = \{0, \frac{\pi}{6}, \frac{\pi}{4}, \frac{\pi}{3}, \frac{\pi}{2}\} \oplus k\frac{\pi}{2}$, com $k \in \{-4, -3, -2, -1, 0, 1, 2, 3\}$, é o conjunto

de pontos de referência no intervalo $[-2\pi, 2\pi]$, que podem ser relacionados com os ângulos fundamentais, e servirá de conjunto auxiliar para a escolha dos extremos do intervalo I.

No método implementado para criar o intervalo I, começamos por resolver a equação/inequação $E \trianglelefteq k$, em $\mathbb{R}$. Para tal, o programa usa o resolutor. Passamos depois à análise da solução obtida.

Se o conjunto-solução for o conjunto vazio ou $\mathbb{R}$, geramos um intervalo completamente aleatório a partir dos pontos de P.

Se o conjunto-solução for $S = A \oplus p\mathbb{Z}$, aplicamos expansão, para que $A' = A \oplus p\mathcal{K}$ contenha o intervalo $[-p, p]$ e, dessa forma, conseguir ter uma maior diversidade de pontos na interseção de A' com o conjunto de pontos de referência P. Como o conjunto de pontos de referência P foi definido no intervalo $[-2\pi, 2\pi]$, se o período p de S for maior do que 2π, os valores referentes aos ângulos fundamentais perdem o sentido porque os pontos de referência cobrem apenas uma fração de A' e corríamos o risco de perder parte da amostra A'. Então, para mapear o conjunto P para $[-p, p]$ definimos o coeficiente $\ell = \frac{p}{2\pi}$ e multiplicamos ℓ por cada elemento do conjunto, resultando assim no conjunto P'. Se p for menor que 2π, não temos este problema e teríamos $P' = P$ no resto do algoritmo para escolha de I.

De seguida, há que decidir se o problema terá ou não solução em I. Através de parâmetros, podemos indicar a probabilidade de cada caso, definindo-os, por exemplo, como 80% para existência de solução em I e 20% no caso contrário.

No caso de querermos um intervalo em que a solução é não vazia, procuramos um intervalo $A'' \subseteq A'$. Para o tipo de exercícios que queremos suportar, sabemos que A' pode ser um conjunto de pontos isolados ou um conjunto de intervalos limitados (os extremos não são ∞ nem $-\infty$). Se for constituído por intervalos limitados, considerámos quatro possibilidades para a escolha de A'':

- A'' é um elemento aleatório de A';
- A'' é a união de dois elementos aleatórios que são consecutivos em A';
- A'' é um subconjunto de um elemento aleatório de A';
- A'' é a união de dois subconjuntos de elementos aleatórios consecutivos de A'.

Cada uma destas opções tem uma probabilidade associada, que é definida pelo utilizador como parâmetro da geração.

Na terceira e quarta opção, escolhemos um subconjunto de um elemento de A'.

Para tal, dividimos o elemento escolhido em partes. Se o elemento de A' escolhido for $E = \texttt{interval}(L'(e_1), R'(e_2))$, calculamos

$$\Delta = \frac{e_2 - e_1}{i},$$

sendo i gerado aleatoriamente no conjunto $\{2, 3, 4\}$. Se $i = 3$, para gerar um subconjunto, basta-nos escolher dois pontos consecutivos no conjunto $\{e_1, e_1 + \Delta, e_1 + 2\Delta, e_1 + 3\Delta\}$, por exemplo.

Depois de termos escolhido A'', estendemos os extremos para um dos pontos em P'. Sendo $A'' = \texttt{interval}(L''(s_1), R''(s_2))$, considerámos que os pontos a e b seriam os segundos mais próximos de s_1 e s_2 no conjunto P', tal que $a < s_1$ e $b > s_2$.

Caso não seja possível, por inexistência dessas posições no conjunto, por exemplo se $s_1 = -p$ não existem números menores em P', escolhemos o ponto imediatamente a seguir ou, em último caso, $a = s_1$ ou $b = s_2$.

Quando A' é um conjunto de pontos isolados, considerámos as seguintes opções para A'':

1. A'' é formado a partir de apenas um elemento aleatório de A';
2. A'' é formado a partir de dois elementos aleatórios de A', tal que os elementos de A' sejam imediatamente próximos.

Se $A'' = \{\texttt{i}(s_1)\}$, estendemos A'' para criar o intervalo I. O processo semelhante ao descrito anteriormente. Os pontos a e b são os segundos mais próximos de s_1 no conjunto P', tal que $a < s_1$ e $b > s_1$. Caso não seja possível, são escolhidos os pontos imediatamente mais próximos ou, em último caso, ou a ou b fica igual a s_1.

No caso de querermos apresentar um intervalo onde não existiria solução, determinamos o conjunto $(A')^C$ complementar de A' em $[-p, p]$, isto é, $(A')^C = [-p, p] \setminus A'$. O conjunto $(A')^C$ pode conter pontos isolados, por exemplo, no caso da restrição $\sin(x) < 1$, que tem solução $]-3\pi/2, \pi/2[\oplus 2\pi\mathbb{Z}$. Neste caso ficaria $(A')^C = \{-3\pi/2, \pi/2\}$ e como pontos isolados não permitem a captação de um intervalo que apresente solução vazia, são retirados e se isso resultar num conjunto vazio (que é o que acontece neste caso em concreto), descartamos esse exercício por completo e passamos à criação do seguinte, usando a próxima expressão no ficheiro.

Se $(A')^C$ for composto por intervalos, considerámos que A'' podia ser:

1. A'' é um elemento aleatório de $(A')^C$;
2. A'' é um subconjunto de um elemento aleatório de $(A')^C$

Para o primeiro caso, definimos I como $A'' = \texttt{interval}(L''(s_1), R''(s_2))$, ou seja, o elemento escolhido de $(A')^C$ ficará como intervalo para o exercício, não sendo alterado. Para a segunda opção reduzimos o alcance de A'' para valores de P'. Nesta redução, seguimos uma abordagem semelhante à aplicada para o alargamento explicado anteriormente, mas os pontos a e b são os segundos mais próximos tal que $s_1 < a$ e $s_2 > b$. Se esta condição não for satisfazível, estaríamos a incluir pontos da solução do problema e por isso reduzimos as posições para os imediatamente mais próximos ou retornamos o estado atual de A''.

Diversificando I, aumentamos o interesse do exercício, pois a mesma restrição com domínios diferentes pode ter resposta diferente. Daí que existam várias possibilidades de escolha aleatória nos vários passos do método.

Capítulo 5

Resolução de problemas

A arquitetura modular do sistema AGILMAT permite que a geração de exercícios e a resolução de equações e inequações funcionem de forma virtualmente isolada. Dizemos virtualmente porque o resolutor tem ação no momento de geração de exercícios, mas o contrário não acontece. Por isso, podemos ver o resolutor como um sistema independente que, para a versão mais restrita em que trabalhámos, pode ser aplicado para determinar domínios ou resolver equações e inequações.

O resolutor permite tratar tipos de restrições diferentes das que apresentámos no Capítulo 4. Todavia, no presente capítulo, focamos as que referimos na Tabela 4.7.

A resolução de restrições $E \trianglelefteq k$ pode ser resumida a duas fases: o cálculo do domínio da expressão E e a aplicação das regras para obter a solução de $E \trianglelefteq k$ no domínio pretendido. Para tal, no sistema AGILMAT, são aplicados dois predicados: `domain_expr/4`, que calcula o domínio `Domf` de uma expressão `E` para um universo `Dom`; e `solving/6` que determina o conjunto `Sols` das soluções num subconjunto do domínio de `E`, que pode ser o domínio.

5.1 Cálculo de domínios

Na implementação do predicado `domain_expr/4`, procurou-se que o mesmo traduzisse regras usuais para cálculo de domínios, lecionadas no Ensino Secundário, como por exemplo:

- o domínio da identidade em $\mathcal{D}$ é $\mathcal{D}$;
- o domínio de $x \rightsquigarrow f(g(x))$ em $\mathcal{D}$, para f polinomial, é o domínio de g em $\mathcal{D}$;
- se n é par, o domínio de $x \rightsquigarrow \sqrt[n]{g(x)}$ em $\mathcal{D}$, é o conjunto solução de $g(x) \geq 0$, para x pertencente ao domínio de g em $\mathcal{D}$;

e que se podem codificar em Prolog assim:

```
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :- !, domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :- !,
 (even(N) ->
    ( domain_expr(U,X,Dom,DomU), solve(DomU,U,geq,rat(0,1),X,Domf) );
    domain_expr(U,X,Dom,Domf)).
```

Por exemplo, se $E = \sqrt{x}$ e $\mathcal{D} =]-\infty, 3]$, a chamada daria $\mathcal{D}_E = [0, 3]$. Mas, para $\mathcal{D} = \mathbb{R}$, daria o domínio $\mathbb{R}_0^+ = [0, +\infty[$.

Nesta secção, descrevemos as regras para tratamento de funções trigonométricas, que implementámos para `domain_expr` suportar tais funções.

Seja $\mathcal{D}$ o universo a considerar. Como visto no Capítulo 4, sabemos que todas as funções trigonométricas estão compostas com alguma polinomial $p(x)$ de primeiro grau. Por isso, para expressões do tipo *trig* ou `sumtrig` ou `sumtrigAB`, os seus argumentos não restringem o domínio da função, e procedemos do modo seguinte:

- Sempre que E for $\sin(p(x))$ ou $\cos(p(x))$, ou quando tiver tipo `sumtrigAB`, é retornado $\mathcal{D}$, porque o seno e cosseno têm domínio $\mathbb{R}$.
- Se E for $\tan(g(x))$, que se define por

  $$\tan(p(x)) = \frac{\sin(p(x))}{\cos(p(x))},$$

  temos de excluir de $\mathcal{D}$ as soluções de $\cos(p(x)) = 0$.
- Se o tipo for `sumtrig` e a expressão não contiver a função tangente, mas apenas seno ou cosseno, retornamos $\mathcal{D}$. Porém, se tivermos duas funções tangente, calculamos o domínio de cada uma no universo $\mathcal{D}$ e o resultado é a interseção desses domínios, visto que o domínio da função soma é a interseção dos domínios das funções envolvidas.

Para as expressões do tipo *vtrig* (apelidamos de *variantes*), que resultam de composições com raízes, potências e valor absoluto, as regras já estavam anteriormente definidas no sistema AGILMAT para o predicado `domain_expr/4`, como ilustrámos acima para $E = \texttt{rad}(U, N)$.

Essencialmente, se $E = \texttt{pow}(U, N)$ ou $E = \texttt{abs}(U)$ ou $E = \texttt{rad}(U, N)$, com N ímpar, o domínio é dado pelo domínio da expressão U no universo $\mathcal{D}$, porque as funções potência, valor absoluto e raíz

de índice ímpar têm domínio $\mathbb{R}$. Se $E = \texttt{rad}(U, N)$, com N par, que corresponde a $E = \sqrt[N]{g(x)}$ será necessário restringir ao conjunto de solução de $g(x) \geq 0$ no domínio de g restrito a $\mathcal{D}$, isto é em $\mathcal{D}_{g(x)}$.

Em suma, se $\mathcal{D}_E$ designar o domínio de $E = f(g(x))$ no universo $\mathcal{D}$, e assumindo que $solver(Expr, \mathbb{U})$ retorna o conjunto solução de $Expr$ em $\mathbb{U}$, temos, com algum abuso de notação:

$$\mathcal{D}_E = \begin{cases} \mathcal{D} & \text{se} \quad f \in \{\texttt{sin},\ \texttt{cos},\ \texttt{sumtrigAB}\} \\ \mathcal{D}_{g(x)} & \text{se} \quad E \in \{|g(x)|,\ \sqrt[n]{g(x)}, (g(x))^m\},\ n \text{ ímpar},\ m \in \mathbb{N} \\ solver(g(x) \geq 0, \mathcal{D}_{g(x)}) & \text{se} \quad E = \sqrt[N]{g(x)},\ N \text{ par} \\ \mathcal{D} \setminus solver(\cos(g(x)) = 0, \mathbb{R}) & \text{se} \quad E = \tan(g(x)) \\ \mathcal{D}_{u(g(x))} \cap \mathcal{D}_{v(h(x))} & \text{se} \quad E \text{ for } \texttt{sumtrig(}g(x)\texttt{,}h(x)\texttt{,_,_,[}u\texttt{,}v\texttt{])} \end{cases}$$

5.2 Resolução de equações e inequações

Como referimos antes, a chamada ao predicado `solving/6`, com `solving(E,Op,K,X,D,Sols)`, determina o conjunto `Sols` das soluções, onde $E \trianglelefteq k$, no domínio $\mathcal{D}$, que se supõe já ser um subconjunto do domínio de E em $\mathbb{R}$. O argumento `Op` denota o operador $\trianglelefteq$ que define a restrição a resolver.

Como acima, na descrição que apresentamos agora, referimos este predicado como *solver*, que retorna `Sols`, dado $E \trianglelefteq k$ e $\mathcal{D}$.

Segundo o modelo usual de execução do Prolog, `solving/6` aplica recursivamente regras, que vão decompondo E até chegar a uma forma terminal, que pode ser a identidade. Tal corresponde a definir a função *solver* como uma função recursiva, que se tivermos, por exemplo, a restrição $\sqrt{x+1} = 1$, poderia envolver a lista de chamadas seria

$$\begin{aligned} &\text{Chamada 1} \quad solver(\sqrt{x+1} = 1, D) \\ &\text{Chamada 2} \quad solver(x + 1 = 1^2, D) \\ &\text{Chamada 3} \quad solver(x = 0, D) \end{aligned}$$

Para cada chamada são aplicadas as regras dependentes não só da expressão como também do operador e do valor de k.

Como cada tipo de restrição pode ter um método próprio de resolução associado. Para as restrições trigonométricas, consideramo-las segundo três grupos: funções trigonométricas simples, somas entre funções trigonométricas e variantes.

5.2.1 Funções trigonométricas simples

Nesta secção, focamos os métodos implementados para resolver equações e inequações trigonométricas da forma $f(ax+b) \trianglelefteq k$, com $a, b \in \mathbb{R}$, $k \in (k_{\texttt{sin}} \cup k_{\texttt{tan}})$, e $\trianglelefteq \in \{=, \neq, <, \leq, >, \geq\}$.

A primeira abordagem passa por perceber que uma restrição do tipo $(f \circ g)(x) \geq k$, para x num domínio I que é subconjunto do domínio de $(f \circ g)$, pode ser traduzida como $(f \circ g)(x) \in [k, +\infty[$ para x num domínio I.

Se conhecermos as funções envolvidas, podemos usar informação sobre o contradomínio de f para simplificar o problema, como observado em [37].

Por exemplo, se tivermos $(f \circ g)(x) \geq 0$ e soubermos que o contradomínio de f é $]-\infty, 0[$ podemos concluir que não existe y tal que $f(y) \geq 0$ tenha solução. Consequentemente, não existe x tal que $(f \circ g)(x) \geq 0$ tenha solução e por isso a solução seria vazia.

Em contrapartida, se for $(f \circ g)(x) \geq -6$ e soubermos que o contradomínio de f é $[0, +\infty[$, então para todo y tal que $y = g(x)$ teríamos $f(y) \geq -6$. Logo, nesse caso, a solução para x seria I.

Como a função seno e cosseno têm um comportamento muito similar, separamos o problema, considerando as funções seno e cosseno num grupo e a função tangente noutro.

5.2.1.1 Seno e Cosseno

Se $f \in \{\sin, \cos\}$, com contradomínio $[-1, 1]$, podemos dar resposta de imediato com a análise sobre o operador $\trianglelefteq$ e o k. Para isso, recorremos à noção de *constraining set* associado a uma restrição simples [37], dada por $\texttt{ctrSet}(z \trianglelefteq k) = S^k_{\trianglelefteq}$, em que $S^k_{\trianglelefteq}$ é definido por $S^k_{\trianglelefteq} = \{z \mid z \trianglelefteq k\}$.

Por exemplo, $S^k_{=} = \{k\}$ ou $S^k_{>} =]k, +\infty[$. No caso da inequação $\sin(x) \geq 1$, temos $S^1_{\geq} = [1, +\infty[$. Designando o contradomínio de uma função f por CD_f, para $f = \sin$, teríamos $S^1_{\geq} \cap CD_{\sin(x)} = \{1\}$. Isso significa que podemos simplificar a restrição a resolver, convertendo o operador $\geq$ para $=$, dado que não existem soluções em $]1, +\infty[$.

Sendo assim, para $f(g(x)) \trianglelefteq k$, com $f = \{\sin, \cos\}$, num domínio D, teríamos como conjunto-solução:

- $\emptyset$, se $S^k_{\trianglelefteq} \cap CD_f = \emptyset$;
- D, se $CD_f \subseteq S^k_{\trianglelefteq}$;
- o que resultasse da aplicação do método direto de resolução $(f(g(x)) = k$, se $S^k_{\trianglelefteq} \cap CD_f = \{k\}$;

- o que resultasse da aplicação do método direto de resolução de restrições ($f(g(x)) \trianglelefteq k$.

Na implementação dos métodos diretos, que se baseiam na análise do comportamento das funções sin e cos em $\mathbb{R}$, começamos por verificar se $f = \cos$, isto é cosseno. Se for, traduzimos em seno (sin), sabendo que $\cos x = \sin\left(\frac{\pi}{2}\right)\cos x + \cos\left(\frac{\pi}{2}\right)\sin x = \sin\left(\frac{\pi}{2} + x\right)$. Desta forma, reduzimos o problema apenas a uma função. Como apenas pretendemos obter a solução final, não tendo focado uma apresentação de soluções passo-a-passo, esta transformação é pedagogicamente admissível.

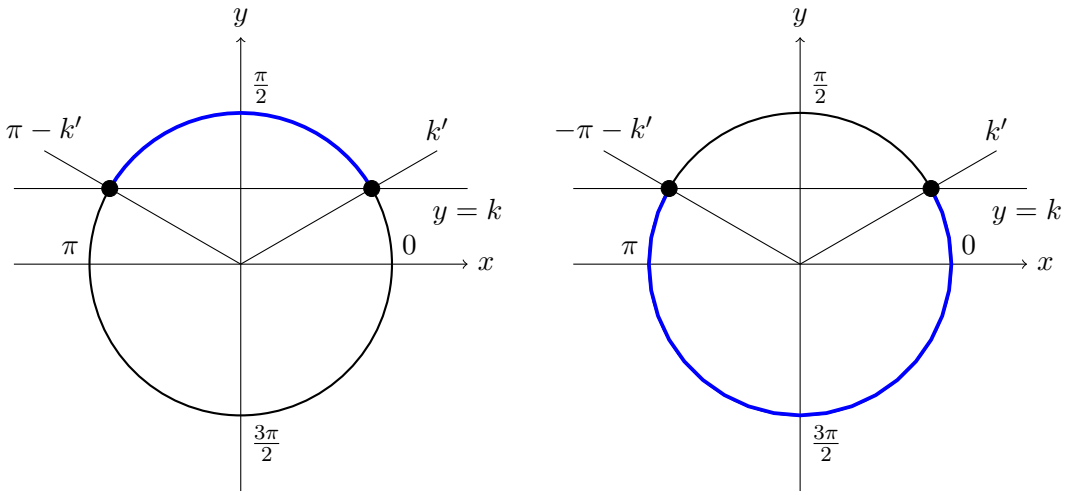


Figura 5.1: Análise de $\sin x \trianglelefteq \sin k'$ no círculo trigonométrico, quando operador $\trianglelefteq$ é $\geq$ ou é $\leq$.

Embora não tivessemos referido este facto no Capítulo 4, na geração de expressões $f(ax+b)$, com $f = \sin$ ou $f = \cos$, o coeficiente b é gerado como $b'\pi$, sendo $b' \in \mathbb{Q}$, porque é a forma mais usual nos livros de referência.

Quando $b' \in \mathbb{Z}$, tal significa que a restrição pode ser ainda simplificada pois, se b' for par, a parcela $b'\pi$ corresponde a dar voltas completas ao círculo. Sendo f uma função periódica, com período 2π, temos $f(ax+b) = f(ax)$, se $b' = 2n\pi$, para algum $n \in \mathbb{Z}$. Se b' for ímpar, então $\sin(ax+b) = \sin(ax+\pi) = -\sin(ax)$, pelo que podemos reescrever, alterando $\trianglelefteq$ e k para os seus simétricos.

Em suma,

- $\sin(ax + b'\pi) \trianglelefteq k$ re-escreve-se em $\sin(ax) \trianglelefteq k$, se $b' \in \mathbb{Z}$ e b' é par;
- $\sin(ax + b'\pi) \trianglelefteq k$ re-escreve-se em $\sin(ax) \trianglerighteq -k$, se $b' \in \mathbb{Z}$ e b' é ímpar

e, nos restantes casos, é aplicado o método de resolução direta.

Se tivermos $\sin(x) = k$ onde $k \in \{-1, 1\}$, conseguimos reduzir a solução apenas a um elemento porque existe apenas uma solução no intervalo $[0, 2\pi[$ para ambos os casos. Já se $k \in]-1, 1[$ existem

sempre dois pontos possíveis, nesse intervalo (como indicado na Figura 5.2).

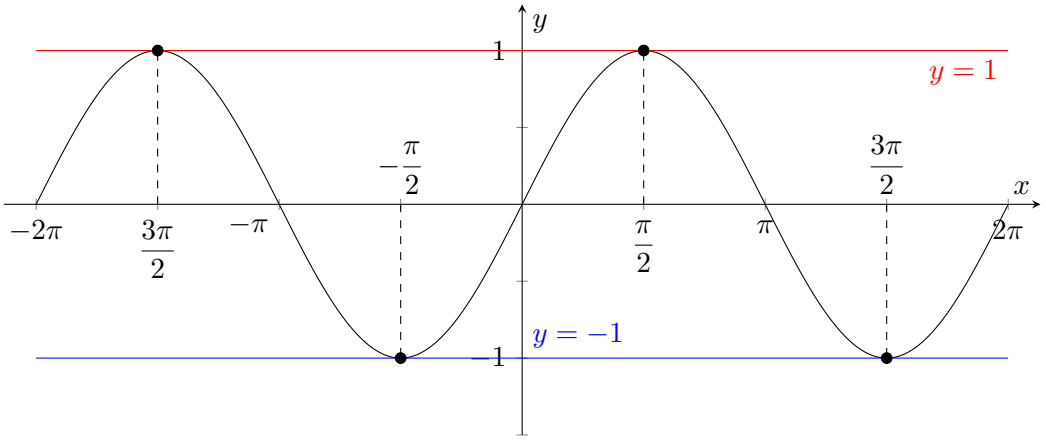


Figura 5.2: Exemplo do comportamento de $\sin(x)$ no intervalo $]-2\pi, 2\pi[$.

As soluções de $\sin(ax+b) \trianglelefteq k$ são dadas pelas regras seguintes, onde $h(x) = \dfrac{x-b}{a}$ e $k = \sin k'$.

$$\begin{cases}
D \cap \left( ]h(k'), h(\pi - k')[+ \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } > \\
D \cap \left([h(k'), h(\pi - k')] + \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } \geq \\
D \cap \left( ]h(-\pi - k'), h(k')[+ \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } < \\
D \cap \left([h(-\pi - k'), h(k')] + \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } \leq \\
D \cap \left(\{h(k'), h(\pi - k')\} + \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } = \wedge\ k \in]-1, 0[\cup]0, 1[\\
D \cap \left(\{h(k')\} + \dfrac{2n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } = \wedge\ k \in \{-1, 1\} \\
D \cap \left(\{h(k')\} + \dfrac{n\pi}{a} \right),\ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } = \wedge\ k = 0 \\
D \setminus solver(\sin(ax+b) = k, D) & \text{se } \trianglelefteq \text{ for } \neq,
\end{cases}$$

5.2.1.2 Tangente

Se f for a função tangente, a análise do comportamento em $(]-\frac{\pi}{2}, \frac{\pi}{2}[+ n\pi,\ n \in \mathbb{Z})$, como mostra a Figura 5.3, permite resolver $\tan(ax+b) \trianglelefteq k$. Dado que k é um dos valores conhecidos, podemos admitir que $k = \tan k'$, com $k' \in]-\frac{\pi}{2}, \frac{\pi}{2}[$.

Tal como para o seno, podemos analisar b para tentar simplificar a restrição. Se $b = b'\pi$ e $b' \in \mathbb{Z}$, reduzimos a função a $\tan(ax)$, porque π é o período da tangente tal como vemos na Figura 5.4).

Concluindo, o conjunto solução $\tan(ax+b) \trianglelefteq k$ é obtido pelas regras seguintes, sendo $h(x) = \dfrac{x-b}{a}$

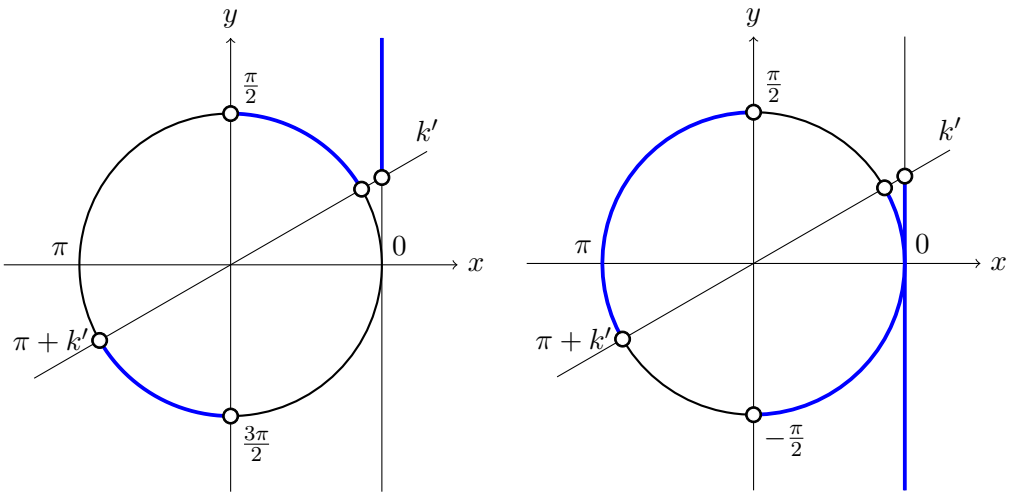


Figura 5.3: Análise de $\tan x \trianglelefteq \tan k'$ no círculo trigonométrico, quando o operador $\trianglelefteq$ é $>$ ou $<$.

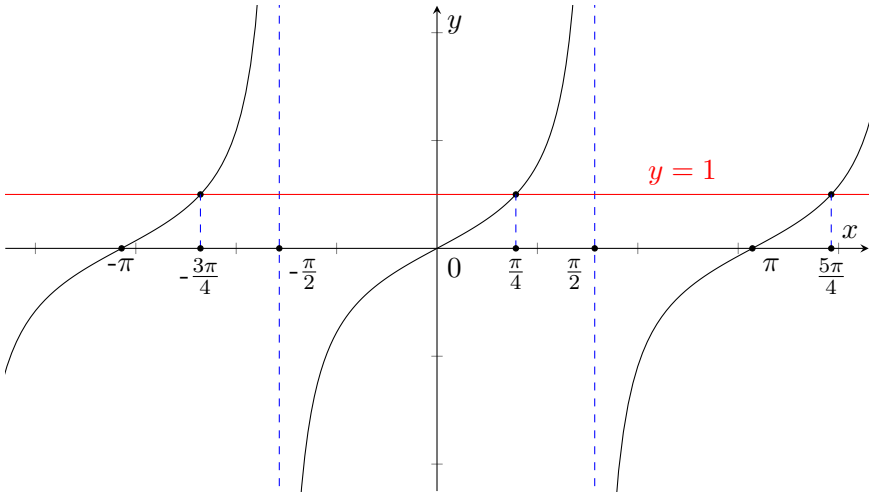


Figura 5.4: Análise da restrição $\tan(x) = 1$.

e $k = \tan k'$.

$$\begin{cases} D \cap \left(\right]h(k'), h\left(\frac{\pi}{2}\right)\left[+ \frac{n\pi}{a}\right), \ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } > \\ D \cap \left(\left[h(k'), h\left(\frac{\pi}{2}\right)\right[+ \frac{n\pi}{a}\right), \ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } \geq \\ D \cap \left(\right]h\left(-\frac{\pi}{2}\right), h(k')\left[+ \frac{n\pi}{a}\right), \ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } < \\ D \cap \left(\right]h\left(-\frac{\pi}{2}\right), h(k')\right] + \frac{n\pi}{a}\Big), \ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } \leq \\ D \cap \left(\{h(k')\} + \frac{n\pi}{a}\right), \ n \in \mathbb{Z} & \text{se } \trianglelefteq \text{ for } = \\ D \setminus solver(\tan(ax+b) = k, D) & \text{se } \trianglelefteq \text{ for } \neq, \end{cases}$$

5.2.2 Somas de Funções Trigonométricas

As somas das funções trigonométricas são dos tipos `sumtrig` e `sumtrigAB`. Sabemos à partida que, num exercício, uma expressão do tipo `sumtrig`, vem sempre associada a um k igual a zero e que nunca é uma soma entre tangente e seno ou cosseno. Se alguma das funções for cosseno convertê-mo-la em seno para podermos tirar partido das regras de soma para produto:

$$\begin{aligned} \sin(\alpha) \pm \sin(\beta) &= 2\sin\left(\frac{\alpha \pm \beta}{2}\right)\cos\left(\frac{\alpha \mp \beta}{2}\right) \\ \tan(\alpha) \pm \tan(\beta) &= \frac{\sin(\alpha \pm \beta)}{\cos(\alpha)\cdot\cos(\beta)} \end{aligned}$$

Se desenvolvermos a primeira regra temos:

$$\begin{aligned} & \sin(ax+b) \pm \sin(cx+d) = 0 \\ \Leftrightarrow \quad & 2\sin\left(\frac{ax+b \pm (cx+d)}{2}\right)\cos\left(\frac{ax+b \mp (cx+d)}{2}\right) = 0 \\ \Leftrightarrow \quad & \sin\left(\frac{a \pm c}{2}x + \frac{b \pm d}{2}\right) = 0 \ \vee \ \cos\left(\frac{a \mp c}{2}x + \frac{b \mp d}{2}\right) = 0 \end{aligned}$$

Tendo simplificado os argumentos das funções, lançamos cada equação para o método de funções simples e reunimos os conjuntos-solução.

No caso da soma das tangentes, a equação resulta em

$$\begin{aligned} & \tan(ax+b) \pm \tan(cx+d) = 0 \\ \Leftrightarrow \quad & \frac{\sin((a \pm c)x + b \pm d)}{\cos(ax+b)\cdot\cos(cx+d)} = 0 \\ \Leftrightarrow \quad & \begin{array}{l} \sin((a \pm c)x + b \pm d) = 0 \ \wedge \\ \cos(ax+b)\cdot\cos(cx+d) \neq 0 \end{array} \end{aligned}$$

Não é necessário resolvermos a desigualdade $\cos(ax+b)\cdot\cos(cx+d) \neq 0$ porque o domínio da expressão já contabiliza este caso. Logo, basta-nos intersetar as soluções de $\sin(ax+b\pm(cx+d)) = 0$ com D.

A resolução de restrições com expressões do tipo `sumtrigAB` já foi um brevemente detalhada na secção 4.2.1 porque a ideia da geração está intrinsecamente ligada à da resolução. Em suma, a

expressão do tipo `sumtrigAB` pode ser reescrita, e a restrição reduz-se a

$$A\sin(f(x)) \mp B\cos(f(x)) \trianglelefteq \frac{k_{\sin}}{\sqrt{A^2+B^2}}$$

$$\Leftrightarrow \quad \frac{A}{\sqrt{A^2+B^2}}\sin(f(x)) \mp \frac{B}{\sqrt{A^2+B^2}}\cos(f(x)) \trianglelefteq \frac{\frac{k_{\sin}}{\sqrt{A^2+B^2}}}{\sqrt{A^2+B^2}}$$

$$\Leftrightarrow \quad \sin(\alpha)\sin(f(x)) \mp \cos(\alpha)\cos(f(x)) \trianglelefteq \frac{k_{\sin}\sqrt{A^2+B^2}}{\sqrt{A^2+B^2}}$$

$$\Leftrightarrow \quad \cos(\alpha \pm f(x)) \trianglelefteq k_{sin}$$

Consequentemente, acabamos por reduzir expressões `sumtrigAB` a `trig` e aplicamos o método para trigonométricas simples.

5.2.3 Variantes

Como para o cálculo dos domínios, a resolução de restrições $|E| \trianglelefteq k$, $\sqrt[N]{E} \trianglelefteq k$ e $E^N \trianglelefteq k$, com E do tipo *vtrig* ou *trig*, são tratadas pelas regras gerais do sistema AGILMAT, anteriormente implementadas.

Valor absoluto Se o resolutor encontrar um caso com valor absoluto, $|g(x)| \trianglelefteq k$, significa que $g(x)$ nunca vai ser negativo. Por consequência, se a restrição tiver $k < 0$ e $\trianglelefteq = \{<, \leq\}$ sabemos automaticamente que a solução é vazia.

Ter imagem sempre positiva significa também que $k > 0$ resulta em ter pelo menos dois conjuntos que têm de ser ou reunidos ou intersetados, dependendo do operador.

Resumindo, as soluções de $|g(x)| \trianglelefteq k$ são obtidas pelas regras seguintes, onde $\trianglerighteq$ representa o operador inverso de $\trianglelefteq$ (por exemplo, se $\trianglelefteq = \{\geq\}$ indica que $\trianglerighteq = \{\leq\}$).

$$\begin{cases} \emptyset & \text{se} \quad (k < 0 \wedge \trianglelefteq = \{=, <, \leq\}) \vee (k = 0 \wedge \trianglelefteq = \{<\}) \\ D & \text{se} \quad (k < 0 \wedge \trianglelefteq = \{>, \geq, \neq\}) \vee (k = 0 \wedge \trianglelefteq = \{\geq\}) \\ solver(g(x) = 0, D) & \text{se} \quad k = 0 \wedge \trianglelefteq = \{=, \leq\} \\ solver(g(x) \neq 0, D) & \text{se} \quad k = 0 \wedge \trianglelefteq = \{\neq\} \\ solver(g(x) \trianglerighteq -k, D) \cup solver(g(x) \trianglelefteq k, D) & \text{se} \quad k > 0 \wedge \trianglelefteq = \{=, >, \geq\} \\ solver(g(x) \trianglerighteq -k, D) \cap solver(g(x) \trianglelefteq k, D) & \text{se} \quad k > 0 \wedge \trianglelefteq = \{\neq, <, \leq\} \end{cases}$$

Raízes Se tivermos no caso de uma restrição $\sqrt[N]{g(x)} \trianglelefteq k$, importa analisar o expoente para perceber se a expressão é sempre não negativa, ou se pode tomar também valores negativos. Recordar que já se assume que o conjunto em que se pretende resolver a restrição é um subconjunto do domínio da expressão, previamente validado.

Porém, se N for par só temos soluções não vazias quando $g(x) \geq 0$, o que implica que, se $g(x) = \sin(x)$ e estivermos a analisar o intervalo $]-2\pi, 2\pi[$, só vamos ter soluções nos intervalos $]-2\pi, -\pi] \cup [0, \pi]$.

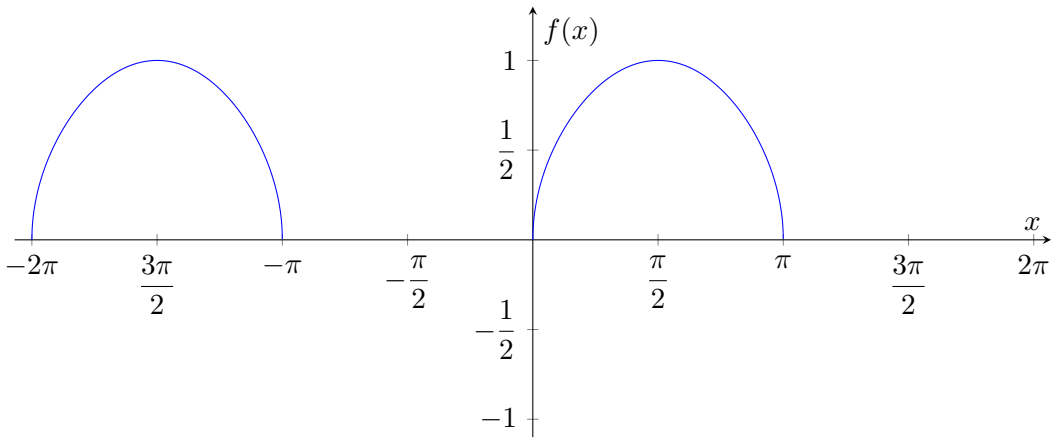


Figura 5.5: Comportamento da função $\sqrt{\sin(x)}$ no intervalo $]-2\pi, 2\pi[$.

Se N for ímpar:

$$solver(\sqrt[N]{g(x)} \trianglelefteq k, D) = solver(g(x) \trianglelefteq k^N, D)$$

Se N for par, $solver(\sqrt[N]{g(x)} \trianglelefteq k, D)$ é

$$\begin{cases} \emptyset & \text{se } (k < 0 \wedge \trianglelefteq = \{=, <, \leq\}) \vee (k = 0 \wedge \trianglelefteq = \{<\}) \\ D & \text{se } (k < 0 \wedge \trianglelefteq = \{>, \geq, \neq\}) \vee (k = 0, \trianglelefteq = \{\geq\}) \\ solver(g(x) = 0, D) & \text{se } k = 0 \wedge \trianglelefteq = \{=\} \\ solver(g(x) \neq 0, D) & \text{se } k = 0 \wedge \trianglelefteq = \{>, \neq\} \\ solver(g(x) \trianglelefteq k^N, D) & \text{caso contrário} \end{cases}$$

Potências As restrições que envolvem potências também é útil analisar a paridade do expoente, porque quando N é par, a função não pode tomar valores negativos. No caso de N ser par, podemos recorrer à função valor absoluto para simplificar o resolutor, como se indica a seguir.

Se N for par $solver(g(x)^N \trianglelefteq k, D)$ é

$$\begin{cases} solver(|g(x)| \trianglelefteq \sqrt[N]{k}, D) & \text{se } k \geq 0 \\ \emptyset & \text{se } \trianglelefteq = \{=, \leq, <\} \\ D & \text{caso contrário} \end{cases}$$

No entanto, se N for ímpar, seria

$$solver(g(x)^N \trianglelefteq k, D) = solver(g(x) \trianglelefteq \sqrt[N]{k}, D)$$

5.3 Limites do resolutor

O AGILMAT teve desde início o objetivo de apresentar apenas soluções exatas para os problemas que resolve. No entanto, esta questão complica-se quando aumentamos o número de tópicos contemplados, porque alguns problemas não podem ser resolvidos computacionalmente e outros requerem métodos mais complexos do que os lecionados no Ensino Secundário.

Daí que a extensão realizada tenha restringido o tipo de expressões suportadas às que constam da Figura 4.7. Para além de serem os tipos de expressões mais comuns em exercícios, testam precisamente o conhecimento das regras ensinadas no Ensino Secundário. Tendo isto em conta, limitámos o resolutor em duas vertentes: lidar apenas com funções trigonométricas cujo argumento é uma polinomial de grau um e limitar k para originar um valor conhecido.

Apesar de não fazerem parte dos objetivos base deste módulo o gerador com um *Rate* elevado pode criar expressões complexas. Por exemplo, com tipos `tan ∘ p1 ∘ x*sin ∘ p1 ∘ x/p1 ∘ x` e `tan ∘ p1 ∘ x*p1 ∘ x/sin ∘ p1 ∘ x`.

Se o exercício for para calcular domínios, não há complicações. Se mudarmos para restrições com qualquer $\trianglelefteq$ e $k \neq 0$, não teremos, em geral, forma de obter a solução.

Há que compreender que não é necessário (nem possível) ter um sistema de geração completo, mas que é possível criar algumas formas que podem ser úteis, como as que apresentámos.

Capítulo 6

Conclusão

Esta tese teve o propósito de adicionar um módulo que gera e resolve exercícios trigonométricos ao protótipo AGILMAT. Este módulo permite estar um passo mais perto de englobar os temas abordados no ensino secundário.

Principiámos o desenvolvimento por nos familiarizarmos com a abordagem explorada no projeto AGILMAT, e por converter alguns módulos do protótipo de SICStus Prolog para SWI-Prolog. A conversão permitiu que o projeto não ficasse dependente de uma licença comercial (ou académica).

O passo seguinte, passou por investigar o formato dos exercícios que envolvem trigonometria no ensino secundário, para assegurar os objetivos fundamentais do AGILMAT. De forma a termos uma visão de como o mercado se desenvolveu nesta área, comparámos a representação de soluções dos manuais escolares com a das plataformas disponíveis gratuitamente.

Tendo concluído o período de investigação, iniciámos a fase de implementação. O seu desenvolvimento reflete-se nos capítulos 3, 4, e 5. As operações entre conjuntos periódicos foram o primeiro ponto a ser abordado. Este objetivo foi o mais complicado dos três, uma vez que requereu muita atenção aos detalhes de cada operação.

Seguiu-se a geração de expressões, onde definimos um conjunto de tipos de expressões. Foi considerada a hipótese das somas trigonométricas serem integradas na categoria *sumexpr* da gramática, mas como esta já é bastante complexa e vasta, contivemos a trigonometria na categoria *vtrig*, com as somas *sumtrig* e *sumtrigAB*. Com isto, tivemos o cuidado de introduzir os novos tipos na gramática de forma a que a sua compreensão fosse intuitiva.

Na geração de exercícios pusemos em prática a investigação feita para alinharmos os exercícios que geramos com o que alunos e professores estão habituados a trabalhar e aplicámos métodos de aleatorização com parâmetros para podermos ter exercícios diferentes mas com o controlo do

utilizador.

O último objetivo a ser concretizado foi a resolução de exercícios. Sabíamos à partida que limites o resolutor ia enfrentar, uma vez que já tinham sido indicados e discutidos em versões anteriores. No entanto, o lado positivo, é que são também limites para o grau de dificuldade que queremos atingir. Portanto, não constituiu um impedimento na implementação.

Em suma, do ponto de vista de resolução de exercícios, o nosso trabalho tem muita dificuldade em competir com a potência de CAS devido ao seu constante desenvolvimento. No entanto, o objetivo principal do AGILMAT é a capacidade de gerar exercícios consoante as preferências do utilizador sem ter de usar uma base de dados. Por isso, destacamos a geração de exercícios de trigonometria sob intervalos que enquadram a solução com base em restrições do utilizador. Mesmo que a expressão seja a mesma, o intervalo associado ao exercício será diferente, muito provavelmente.

Apesar de termos concretizado o módulo de trigonometria para o protótipo, ainda há espaço para aperfeiçoamentos e adições. Deixamos aqui algumas questões a ponderar ou possíveis avanços no âmbito do trabalho que fizemos.

Operações com conjuntos periódicos Na implementação atual, existem alguns problemas na cobertura das operações com conjuntos periódicos. Se calcularmos o domínio de $\sqrt{x} \cdot \tan(x)$ encontramos o conjunto

$$[0, +\infty[\cap \left]-\frac{\pi}{2}, \frac{\pi}{2}\right[\oplus \pi\mathbb{Z},$$

cuja solução pode ser representada como

$$\left(\left[0, \frac{\pi}{2}\right[\cup \left]\frac{\pi}{2}, \pi\right]\right) \oplus \pi\mathbb{Z}_0^+.$$

Se analisarmos do ponto de vista pedagógico, poderá ter interesse suportar este tipo de conjuntos, embora a resolução de restrições trigonométricas se limite muitas vezes a pontos de um intervalo $[a, b]$.

Exercícios Por limite de tempo, o tipo `sumtrig` não tem expressões entre seno e tangente nem cosseno e tangente. Com a introdução deste caso, asseguramos a combinatória total entre seno, cosseno e tangente.

Partimos do princípio que os problemas seguem as mesmas restrições para exercícios do tipo `sumtrig` atual ($\trianglelefteq = \{=\}$ e $k = 0$). Para o caso $\sin(\alpha) + \tan(\alpha)$, a resolução de exercícios de restrições não teria problemas porque o problema traduz-se em

$$\frac{\sin(\alpha)(\cos(\alpha) + 1)}{\cos(\alpha)} = 0 \Leftrightarrow \sin(\alpha) = 0 \vee \cos(\alpha) + 1 = 0$$

e os nossos métodos aritméticos e de conjuntos atuais resolvem este caso. Já $\cos(\alpha) + \tan(\alpha)$ reflete-se em

$$\frac{\cos^2(\alpha) + \sin(\alpha)}{\cos(\alpha)} = \frac{1 - \sin^2(\alpha) + \sin(\alpha)}{\cos(\alpha)}$$

mas o protótipo não está preparado para mudanças de variável. Bastava que $\sin(\alpha) = \beta$,

$$\frac{1 - \sin^2(\alpha) + \sin(\alpha)}{\cos(\alpha)} = \frac{1 - \beta^2 + \beta}{\cos(\alpha)}$$

e temos uma função quadrática no numerador. Seria possível acrescentar tipos para poder lidar com estes casos.

Funções Trigonométricas Inversas Apesar das funções trigonométricas inversas serem levemente abordadas pelas metas curriculares pelo Ministério da Educação, os manuais escolares apenas possuem exercícios básicos acerca das mesmas.

As funções trigonométricas inversas poderiam ter sido introduzidas no AGILMAT, numa extensão da gramática.

Extensão para AgilmatQuiz O trabalho realizado permite que o AgilmatQuiz tenha agora um recurso de trigonometria para testes de escolha múltipla. Todavia, é necessário desenvolver métodos para a geração de respostas erradas (isto é, de distratores). Na linha da abordagem seguida anteriormente no AgilmatQuiz, seria útil efetuar um levantamento de erros frequentes, que possam ser traduzidos por regras no resolutor, que originarão distratores plausíveis.

Bibliografia

[1] Michael Beeson. Mathpert: Computer support for learning algebra, trigonometry, and calculus. Em: A. Voronkov, editor, *Logic Programming and Automated Reasoning*, volume 624 of *Lecture Notes in Artificial Intelligence*, pages 454–457, 1992.

[2] Michael Beeson. Design principles of mathpert: software to support education in algebra and calculus. Em: Norbert Kajler, editor, *Computer-Human Interaction in Symbolic Computation*, Texts and Monographs in Symbolic Computation, pages 163–177. Springer, 1998. doi:10.1007/978-3-7091-6461-7_9.

[3] A. Bívar, C. Grosso, F. Oliveira, L. Loura, e M. C. Timóteo. Programa e metas curriculares - Matemática A - Ensino Secundário, 2013. Direção-Geral da Educação do Ministério da Educação e Ciência, Lisboa.

[4] Sanjay K. Dwivedi e Vaishali Singh. Research and reviews in question answering system. *Procedia Technol*, 10:417–424, Janeiro 2013. ISSN: 2212-0173. doi:10.1016/j.protcy.2013.12.378.

[5] Martha Alicia Magaña Echeverría, Pedro C. Santana-Mancilla, Hector Fabian Quintero Carrillo, e Enrique Alejandro Fernández Enciso. Natural user interfaces to teach math on higher education. *Procedia - Social and Behavioral Sciences*, 106:1883–1889, Dezembro 2013. ISSN: 1877-0428. doi:10.1016/j.sbspro.2013.12.214.

[6] Xia Feifei e Zhang Guangnian. Design and implementation of a Java-based search engine algorithm analysis system. Em: *4th International Conference on Computer Science & Education*, pages 1040–1043. IEEE, Julho 2009. doi:10.1109/ICCSE.2009.5228477.

[7] Teresa Franklin e Li-Wei Peng. Mobile math: Math educators and students engage in mobile learning. *J. Comput. High. Educ.*, 20(2):69–80, Novembro 2008. ISSN: 1867-1233. doi:10.1007/s12528-008-9005-0.

[8] GeoGebra. GeoGebra, 2001–2022. Online [acedido Agosto 2022].

[9] Giorgi Goguadze. Activemath – generation and reuse of interactive exercises using domain reasoners and automated tutorial strategies, 2011. PhD thesis, Saarland University. doi:10.22028/D291-26097.

[10] Markus Hohenwarter, Judith Hohenwarter, Yves Kreis, e Zsolt Lavicza. Teaching and learning calculus with free dynamic mathematics software GeoGebra. Em: *11th International Conference in Mathematics Education (ICME)*, Monterrey, Mexico, Julho 2008.

[11] Matthew B. Hoy. Wolfphram|alpha: A brief introduction. *Medical Reference Services Quarterly*, 29(1):67–74, 2010. PMID: 20391166. doi:10.1080/02763860903485225.

[12] ILIAS. About ILIAS, 1997–2022. Online [acedido Setembro 2022].

[13] ILIAS. ILIAS open source e-learning, 2018. Brochura [acedido Agosto 2022].

[14] Rui Isidro, Joaquim Sousa Pinto, e António Batel Anjo. SA^{3}C-platform of evaluation system and computer assisted learning. *WEAS Transactions on Advances in Engineering Education*, 1 (2):1–6, 2005.

[15] Bobbie Johnson. Where does Wolfram Alpha get its information? *The Guardian*, Fevereiro 2017. Artigo de divulgação online [acedido Setembro 2022].

[16] Khan Academy. Khan Academy, 2008–2022. Online [acedido Setembro 2022].

[17] Vanda Lourenço, Alexandra Duarte, Alexandra Nunes, Ana Amaral, Conceição Gonçalves, Madalena Mota, e Rosário Mendes. PISA 2018 – Portugal relatório nacional, 2019. IAVE, Online [acedido Setembro 2022].

[18] Dulce Miranda, Escola Secundária de Barcelos, Lia Oliveira, e António Batel Anjo. Um Estudo de Caso com Sistema PMATE (10º ano, Geometria). *V Conferência Internacional de Tecnologias de Informação e Comunicação na Educação*, 2007.

[19] Moodle. About Moodle - MoodleDocs, 2002–2022. Online [acedido Setembro 2022].

[20] Petr Necesal e Jan Pospıšil. Experience with teaching mathematics for engineers with the aid of Wolfram Alpha. Em: *Proceedings of the World Congress on Engineering and Computer Science*, volume 1, pages 271–274, 2012.

[21] Maria Paula Oliveira e Paula Carvalho. Mathematics exercise generator: The language of parameterized exercises. Em: *Developing Technology Mediation in Learning Environments*, pages 264–282. IGI Global, 2020. doi:10.4018/978-1-7998-1591-4.ch015.

[22] J. S. Pinto, M. P. Oliveira, A. B. Anjo, S. I. Vieira Pais, R. O. Isidro, e M. H. Silva. TDmat–mathematics diagnosis evaluation test for engineering sciences students. *International Journal of Mathematical Education in Science and Technology*, 38(3):283–299, 2007. doi:10.1080/00207390601035476.

[23] PORDATA. Insucesso na leitura, matemática e ciências, Novembro 2021. Online [acedido Agosto 2022].

[24] PORDATA. Indivíduos com 16 e mais anos que utilizam computador e Internet em % do total de indivíduos: por grupo etário, Julho 2022. Online [acedido Agosto 2022].

[25] Presidência do Conselho de Ministros. Resolução do Conselho de Ministros n.º 30/2020, 2020.

[26] Adelaide João Proença e Ana Paula Canavarro. Condições de uso do computador nas aulas de matemática do Ensino Secundário em Portugal. *Campo Abierto, Revista de Educación*, 40(1), 2021. ISSN: 0213-9529. doi:10.17398/0213-9529.40.1.119.

[27] Syed A. Raza, Wasim Qazi, Komal Akram Khan, e Javeria Salam. Social isolation and acceptance of the learning management system. *Journal of Educational Computing Research*, 59(2):183, Abril 2021. doi:10.1177/0735633120960421.

[28] Christopher Sangwin e Tim Hunt. Moodle plugins directory: STACK, Setembro 2022. Documentation online.

[29] Soumik Sarkar. A brief history of online education, Maio 2020. Online.

[30] Mariya P. Shyshkina e Maiia V. Marienko. The use of the cloud services to support the math teachers training. *Arnold E. Kiv, Mariya P. Shyshkina*, Julho 2020. ISSN: 1613-0073.

[31] E. C. Silva, J. A. Fernandes, B. D. Silva, e M. R. M. Morelatti. Integração de computadores nas práticas pedagógicas de professores de matemática. *Holos*, 8:303–317, 2016. doi:10.15628/holos.2016.5165.

[32] STACK. STACK 4.4, 2010. Online [acedido Setembro 2022].

[33] STACK. STACK Documentation, 2010. Documentação online [acedido Setembro 2022].

[34] Ana Paula Tomás e José Paulo Leal. A CLP-based tool for computer aided generation and solving of maths exercises. Em: *Practical Aspects of Declarative Languages*, pages 223–240. Springer, Berlin, Germany, Dezembro 2002. doi:10.1007/3-540-36388-2_16.

[35] Ana Paula Tomás e José Paulo Leal. Automatic generation and delivery of multiple-choice math quizzes. Em: *Principles and Practice of Constraint Programming*, pages 848–863. Springer, Berlin, Germany, 2013. doi:10.1007/978-3-642-40627-0_62.

[36] Ana Paula Tomás e Cláudia Carvalho. On representations of periodic sets, 2007. Manuscrito, DCC-FCUP & LIACC, Universidade do Porto.

[37] Ana Paula Tomás, Nelma Moreira, e Nuno Pereira. Designing a solver for arithmetic constraints to support education in mathematics. Em: Ilias Maglogiannis, Kostas Karpouzis, e Max Bramer, editores, *Artificial Intelligence Applications and Innovations, 3rd IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI)*, volume 204 of *IFIP*, pages 433–441. Springer, 2006. doi:10.1007/0-387-34224-9_50.

[38] Ana Paula Tomás, José Paulo Leal, e Marcos Aurélio Domingues. A web application for mathematics education. Em: Howard Leung, Frederick W. B. Li, Rynson W. H. Lau, e Qing Li, editores, *Advances in Web Based Learning - ICWL 2007, 6th International Conference, Edinburgh, UK, August 15-17, 2007, Revised Papers*, volume 4823 of *Lecture Notes in Computer Science*, pages 380–391. Springer, 2007. doi:10.1007/978-3-540-78139-4_34.

[39] Tribunal de Contas. Ensino a distância e digitalização nas escolas durante a pandemia: Uma resposta rápida e adaptada à pandemia, mas limitada pela insuficiência de competências e meios digitais a requerer investimento, 2021. Relatório de Auditoria N.º 9/2021 [acedido Setembro 2022].

[40] Udemy. Udemy, 2010–2022. Online [acedido Setembro 2022].

[41] Wolfram. Future directions of Wolfram|Alpha, Novembro 2009. Online [acedido Setembro 2022].

[42] Stephen Wolfram. Overcoming artificial stupidity, 2012. Online [acedido Setembro 2022].

[43] Wolfram|Alpha. Wolfram|alpha, 2009–2022. Online [acedido Setembro 2022].

[44] Wolfram|Alpha. Wolfram Problem Generator: Online Practice Questions & Answers, 2009–2022. Online [acedido Setembro 2022].

[45] Wolfram|Alpha. New Wolfram Problem Generator: Practice and Learn, Outubro 2013. Online [acedido Setembro 2022].

[46] Liming Zhang e Jianli Jiao. A study on effective Math teaching strategy design in hybrid learning environment. Em: *Hybrid Learning*, pages 212–223. Springer, Berlin, Germany, 2011. doi:10.1007/978-3-642-22763-9_20.