A sequencia da apresentacao deve serguir o seu tipo de trabalho

1 Implementação e Resultados

Aqui ... altere para o contexto de seu problema

Falar do *search* rapidamente .... e enfatizar os doi pontos que serao explorados:

1. Sel Var: Método de seleção das variáveis:
   define a maneira em que as variáveis serão selecionadas pelo solver para receberem um valor;

2. Atr Dom: Método de atribuição dos valores:
   define a maneira em que os valores dentro do domínio são escolhidos e atribuídos às variáveis durante o processo de exploração e busca;

Tabela 1: LEGENDA DA TABELA Eh sobre a tabela MESMO

Sel. Variável / Atr. Dominio	`first_fail`	`anti_first_fail`	`occurrence`
`indomain_min`	...	...	...
`indomain_max`	...	...	...
`indomain_median`	...	...	...
`indomain_split`	...	...	...
`indomain_random`	...	...	...
`indomain_interval`	...	...	...

1.0.1 Solução em EcliPSE

1.0.2 Solução em PR

Aqui ... altere para o contexto de seu problema ECLIPSE

Aqui ... altere para o contexto de seu problema. Qual o objetivo do artigo mesmo? Retome o artigo aqui

Tabela 2: LEGENDA DE FIGURA é EMBAIXO ... aqui eh uma **tabela**

Sel. Variável / Atr. Dominio	`most_constrained`	`max_regret`
`indomain_min`	...	...
`indomain_max`	...	...
`indomain_median`	...	...
`indomain_split`	...	...
`indomain_random`	...	...
`indomain_interval`	...	...

O experimento consistiu em implementar .....

A tabela 1.0.2 apresenta alguns destes valores quando executados em uma máquina padrão com 1.8 GHz de velocidade de CPU, 512 Kbytes de memória principal.

Nao hah como fugir de uma tabela como esta que se segue
ESTA DEVE SER MUITO, MAS MUITO MELHORADA

Tabela 3: Resultados

Tabuleiro	Outras TécnicasProLog	CP	Número de Combinações
4×4	$\cong$ 0.512 ms	$\cong$ 2.979 ms	2
6×6	$\cong$ 195.236 ms	$\cong$ 113.003 ms	4
8×8	$\cong$ 1029.973 ms	$\cong$ 722.416 ms	92
10×10	$\cong$ 99.553 seg	$\cong$ 15.00 seg	724
12×12	$\geq$ 120 min	$\cong$ 334.117 seg	14200

Embora tenha-se obtido tempos aceitáveis, a estatística dos tempos obtidos demonstram a complexidade exponencial deste problema. Para $N > 10$, este problema assume uma complexidade superior a 2^N, logo, um problema NP-completo. Tratando-se de uma otimização, este é um clássico NP-difícil, do inglês, *NP-hard* [**?**]. Com tabuleiros de lado maior que 8 unidades os tempos crescem por um fator exponencial. A partir de um tabuleiro de tamanho 7, inicia as vantagens da utilização da CP comparada com a programação

em lógica tradicional. Mesmo com instâncias pequenas, devido a natureza do problema, logo se começa a notar as diferenças de tempos de execução. Encontrar a melhor combinação não o fator mais complicado neste experimento, mas sim encontrar *todas* combinações.

Em estratégias de buscas por melhoramentos como algoritmos genéticos, *simulated annealing*, são interessantes se o objetivo fosse encontrar uma única solução [**?**]. Contudo, no problema aqui proposto, a complexidade do controle em evitar as soluções duplicadas por estes métodos, deixaria de ser uma abordagem viável neste problema. Assim, a completude quanto as soluções existentes de um dado problema, torna a CP como uma técnica atrativa.