



UNIVERSIDADE FEDERAL DO CEARÁ

Lógica para Computação
Conjunto Dominante de tamanho 3

Relatório final do Trabalho em Equipe apresentado à disciplina Lógica para Computação do curso de Engenharia de Software da Universidade Federal do Ceará.

Professor(a): Robertty Costa

Alunos:

Ruan David da Silva Barros, 553093

Lucas de Souza Almeida, 569990

Russas-CE
2026

Índice

1. Introdução.....	3
1.1. Problema Escolhido.....	3
2. Modelagem do Problema.....	3
2.1. Predicados em Lógica de Primeira Ordem(LPO).....	3
2.2. Representação na Sintaxe TPTP (Vampire).....	4
3. Testes em Grafos Pequenos.....	3
I. Grafo M(Caso Satisffeito).....	3
II. Grafo Isolado:(Caso Insatisffeito).....	5
III. Grafo Triângulo(Caso Satisffeito).....	6
IV. Grafo Disperso(Caso Satisffeito).....	8
4. Teste Final em Grafo Grande.....	10
4.1. Grafo Escolhido	12
4.2. Sintaxe TPTP	13
4.3. Resultados do Teste.....	13
5. Conclusão.....	14

1. Introdução

O presente relatório detalha o desenvolvimento do trabalho final da disciplina de Lógica para Computação, ministrada pelo Professor Robertty Costa no Semestre 2025.2, na Universidade Federal do Ceará – Campus Russas. O objetivo central desta atividade consiste na modelagem e resolução de problemas clássicos da teoria dos grafos utilizando Lógica de Primeira Ordem (LPO).

1.1. Problema Escolhido

O problema abordado neste trabalho é o do **Conjunto Dominante**. Um conjunto dominante em um grafo é um subconjunto de vértices tal que, cada vértice do grafo ou pertence a este conjunto, ou é vizinho de pelo menos um vértice dele. O objetivo específico deste trabalho é verificar a existência de conjuntos dominantes de tamanhos específicos ($k=1$ e $k=3$) usando provas automáticas.

2. Modelagem do Problema

2.1 Predicados em Lógica de Primeira Ordem(LPO)

Para modelar este problema, utilizamos o predicado $\text{adj}(x,y)$ para representar a adjacência entre vértices. A propriedade de um **Conjunto Dominante de tamanho k=3** exige que existam três vértices (c_1, c_2, c_3) tal que, para **todo** vértice X do grafo, X pertença a esse conjunto ou seja vizinho de um de seus membros.

A fórmula lógica correspondente é:

$$\exists c1 \exists c2 \exists c3 \forall x((x = c1 \vee x = c2 \vee x = c3) \vee (\text{adj}(x, c1) \vee \text{adj}(x, c2) \vee \text{adj}(x, c3)))$$

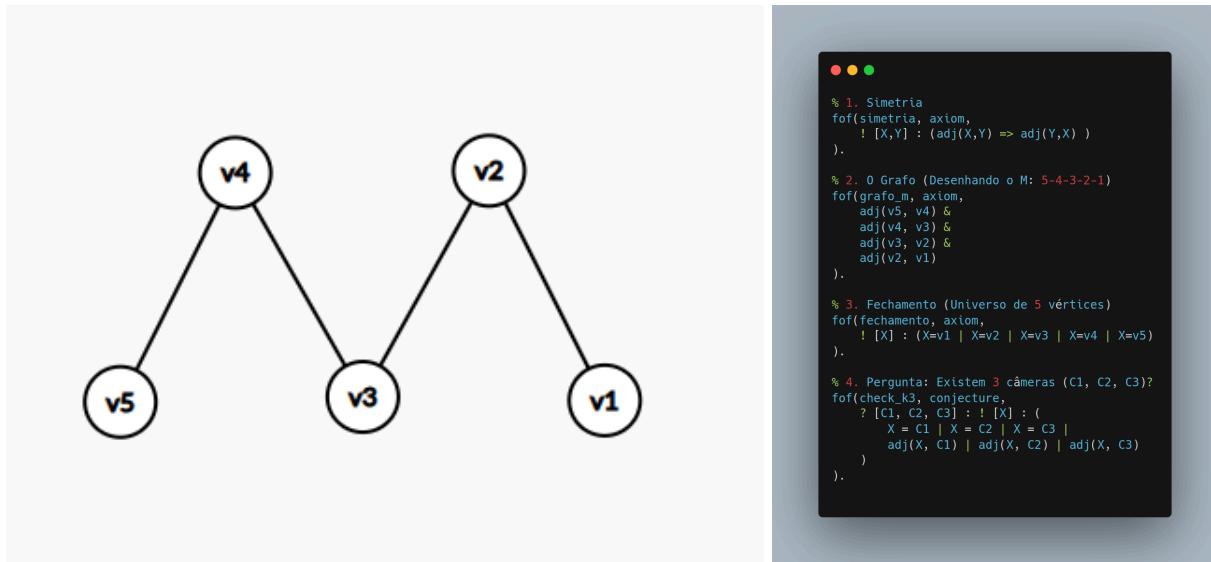
2.2 Representação na Sintaxe TPTP (Vampire)

Representação na Sintaxe TPTP (Vampire) A propriedade foi traduzida para a sintaxe do Vampire. Definimos o grafo através de axiomas de adjacência e um axioma de fechamento de domínio. A conjectura para tamanho 3, utilizada neste trabalho, é escrita como:

```
fof(check_k3, conjecture,
? [C1, C2, C3] : ! [X] : (
    X=C1 | X=C2 | X=C3 |
    adj(X,C1) | adj(X,C2) | adj(X,C3)
)
).
```

3. Testes em Grafos Pequenos

I. Grafo M(Caso Satisffeito)



Código:

```
fof(simetria, axiom,
! [X,Y] : (adj(X,Y) => adj(Y,X) )
```

```

).
fof(graf0_m, axiom,
  adj(v5, v4) &
  adj(v4, v3) &
  adj(v3, v2) &
  adj(v2, v1)
).

```

```

fof(fechamento, axiom,
  ! [X] : (X=v1 | X=v2 | X=v3 | X=v4 | X=v5)
).

```

```

fof(check_k3, conjecture,
  ? [C1, C2, C3] : ! [X] : (
    X = C1 | X = C2 | X = C3 |
    adj(X, C1) | adj(X, C2) | adj(X, C3)
  )
).

```

Resultado:

```

root@Nether:/home/vampire/Irabalho_Final# nano graf0M.p
root@Nether:/home/vampire/Irabalho_Final# ./vampire graf0M.p
% Reading input file ./vampire/graf0M.p
% Refutation found. Thanks to Vampire!
% SZS status Theorem for graf0M
% SZS output start for proof
1. ! [X] : adj((X,X)) = adj(X1,X1) [input/axiom]
2. adj(v5,v4) & adj(v4,v3) & adj(v3,v2) & adj(v2,v1) [input/axiom]
3. ! [X0] : (X0 = v1 | X0 = v2 | X0 = v3 | X0 = v4 | X0 = v5) [input/axiom]
4. ! [X1] : (X1 = v1 | X1 = v2 | X1 = v3 | X1 = v4 | X1 = v5) [input/axiom]
5. -! [X0,X1,X2] : ! [X3] : (X3 = X0 | X3 = X1 | X3 = X2 | adj(X0,X3) | adj(X2,X3) | adj(X3,X2)) [input/conjecture]
6. ! [X0,X1,X2] : ! [X3] : (X3 = X0 | X3 = X1 | X3 = X2 | adj(X0,X3) | adj(X2,X3) | adj(X3,X2)) [negated conjecture 4]
7. ! [X0,X1,X2] : ! [X3] : (X3 = X0 | X3 = X1 | X3 = X2 | adj(X0,X3) & -adj(X3,X1) & -adj(X3,X2)) [enf transformation 5]
8. ! [X0,X1,X2] : ! [X3] : (X3 = X0 | X3 = X1 | X3 = X2 | adj(X0,X3) & -adj(X3,X1) & -adj(X3,X2)) => (sk0(X0,X1,X2) := X1 & sk0(X0,X1,X2) := X2 & sk0(X0,X1,X2) := X1 & sk0(X0,X1,X2),X0) & -adj(sk0(X0,X1,X2),X1) & -adj(sk0(X0,X1,X2),X2)) [skolem symbol introduction]
9. ! [X0,X1,X2] : X0 & sk0(X0,X1,X2) := X1 & sk0(X0,X1,X2) := X2 & -adj(sk0(X0,X1,X2),X0) & -adj(sk0(X0,X1,X2),X1) & -adj(sk0(X0,X1,X2),X2)) [skolem symbol introduction]
10. -adj(X0,X1) | adj(X1,X0) [cnf transformation 6]
11. adj(X0,X1) | adj(X1,X0) [cnf transformation 7]
12. adj(v5,v4) [cnf transformation 8]
13. adj(v5,v4) [cnf transformation 9]
14. adj(v5,v4) [cnf transformation 10]
15. v1 = X0 | v2 = X0 | v3 = X0 | v4 = X0 | v5 = X0 [cnf transformation 3]
17. -adj(X0,X1,X2) | adj(X1,X0,X2) [cnf transformation 9]
18. adj(X0,X1,X2) | -adj(X1,X0,X2) [cnf transformation 9]
19. sk0(X0,X1,X2) := X2 [cnf transformation 9]
20. sk0(X0,X1,X2) := X1 [cnf transformation 9]
21. sk0(X0,X1,X2) := X0 [cnf transformation 9]
22. sk0(X0,X1,X2) := X0 [resolution 18,15]
23. v1 = X0 | v2 = sk0(X0,X1,X2) | v3 = sk0(X0,X1,X2) | v4 = sk0(X0,X1,X2) | v5 = sk0(X0,X1,X2) [superposition 21,15]
24. v5 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) | v4 = sk0(v1,X0,X1) | v2 = sk0(v1,X0,X1) [equality resolution 28]
25. v5 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) | v4 = sk0(v1,X0,X1) | v1 = sk0(v1,X0,X1) [equality resolution 26,62]
26. v2 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) | v4 = sk0(v1,X0,X1) | v5 = sk0(v1,X0,X1) [equality resolution 68]
27. v2 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) | v4 = sk0(v1,X0,X1) | v1 = sk0(v1,X0,X1) [superposition 18,108]
28. -adj(v2,v1) | v4 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) [avatar definition]
29. 11 <-> ! [X0] : (v4 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1)) [avatar component clause 122]
30. v4 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) [forward subsumption resolution 116,11]
31. v1 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) [forward subsumption resolution 116,11]
32. v1 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) [superposition 19,122]
33. v1 = sk0(v1,X0,X1) | v3 = sk0(v1,X0,X1) [superposition 155]
34. adj(v4,v5) :- (11) [superposition 17,183]
35. 198. $False <- (11) [forward subsumption resolution 194,22]
36. 199. -11 [avatar contradiction clause 198]
37. 199. -11 [sat contradiction 199]
38. 199. # [rat si2,si1]
39. 200. $False [avatar sat refutation si8]
% SZS output end Proof for graf0M
% SZS status Theorem for graf0M
% Version: Vampire 5.6.1 (Release build, commit 1b13leaf on 2026-01-18 12:14:59 +0000)
% Linked with Z3 4.8.0 3c47fd96cf5645d8c42b2c819d9e9a843b0aa721 NOTFOUND
% Configuration: -t 1000000000 -t 1000000000
% Termination reason: Refutation
% Time elapsed: 0.014 s
% Peak memory usage: 14 MB
% Instructions burned: 10 (million)
% ...
root@Nether:/home/vampire/Irabalho_Final#

```

Análise do Resultado: Para este teste, utilizou-se um grafo linear de 5 vértices disposto em formato de "M" (v5-v4-v3-v2-v1). A conjectura verificou a existência de um Conjunto Dominante de tamanho k=3.

O provador Vampire retornou o status "**Theorem**" com uma prova por **Refutação** em apenas 0.014s. Este resultado é consistente com a teoria dos grafos: para um caminho de 5 vértices, um conjunto dominante mínimo teria tamanho 2 (por exemplo, vértices v2, v cobrem todo o grafo). Como a conjectura perguntava se era possível cobrir com 3 câmeras, a condição foi satisfeita com folga, resultando em uma prova de sucesso imediata.

II. Grafo Isolado:(Caso Insatisfiço)



Código:

```
fof(simetria, axiom, ! [X,Y] : (adj(X,Y) => adj(Y,X)) ).
```

```
fof(fechamento, axiom,
    ! [X] : (X=v1 | X=v2 | X=v3 | X=v4)
).
```

```
fof(check_k3, conjecture,
    ? [C1, C2, C3] : ! [X] : (
        X = C1 | X = C2 | X = C3 |
        adj(X, C1) | adj(X, C2) | adj(X, C3)
    )
).
```

Resultado:

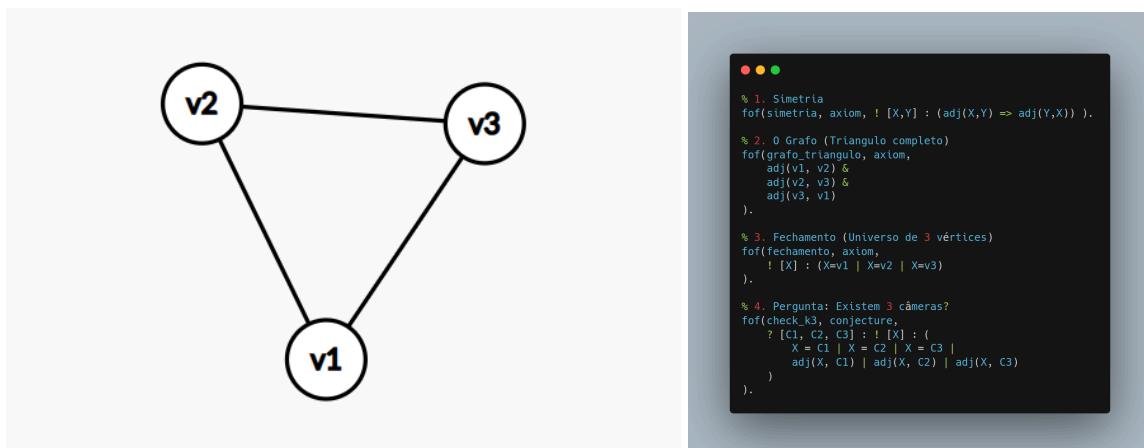
```

root@Nether:/home/vampire/Trabalho_Final# nano linha4.p
root@Nether:/home/vampire/Trabalho_Final# ./vampire linha4.p
% Running in auto input_syntax mode. Trying TPTP
% Time limit reached!
%
% -----
% Version: Vampire 5.0.1 (Release build, commit 1b13eaf on 2026-01-18 12:14:50 +0000)
% Linked with Z3 4.14.0.0 3c47fd96cf5645d0c42b2c819d9e9a84380aa721 NOTFOUND
% CaDiCaL version: 2.1.3
% Termination reason: Time limit
% Termination phase: Saturation
% Time elapsed: 60.001 s
% Peak memory usage: 1093 MB
% Instructions burned: 856478 (million)
%
%
root@Nether:/home/vampire/Trabalho_Final#

```

Análise do Resultado: Para simular um caso de falha (Caso Negativo) utilizando a conjectura padrão de $k=3$, modelou-se um cenário com **4 vértices isolados** (sem arestas conectando-os). A topologia desconexa impõe que cada vértice só pode ser dominado por si mesmo. Logo, para cobrir 4 vértices, seriam estritamente necessárias 4 câmeras. Ao questionar o provador se **3 câmeras** seriam suficientes, o Vampire atingiu o limite de tempo (**Time limit reached**) sem encontrar uma prova de sucesso (Refutation). Este resultado é fundamental para validar a corretude da modelagem, demonstrando que o código não retorna "Sim" indiscriminadamente, mas respeita as restrições lógicas e topológicas do grafo, rejeitando corretamente configurações impossíveis.

III. Grafo Triângulo(Caso Satisffeito)



```
fof(simetria, axiom, ! [X,Y] : (adj(X,Y) => adj(Y,X)) ).
```

```
fof(graf0_triangulo, axiom,
    adj(v1, v2) &
    adj(v2, v3) &
    adj(v3, v1)
).
```

```
fof(fechamento, axiom,
    ! [X] : (X=v1 | X=v2 | X=v3)
).
```

```
fof(check_k3, conjecture,
    ? [C1, C2, C3] : ! [X] : (
        X = C1 | X = C2 | X = C3 |
        adj(X, C1) | adj(X, C2) | adj(X, C3)
    )
).
```

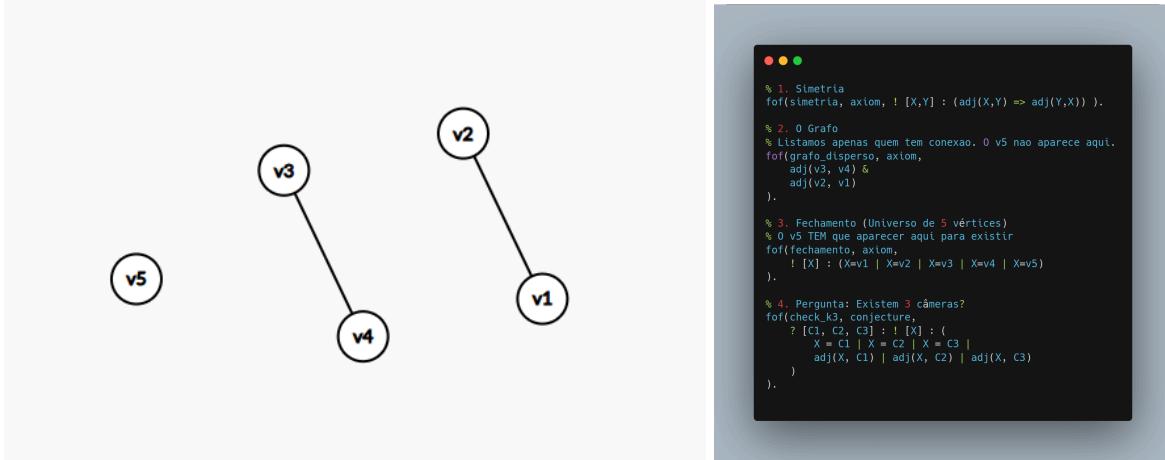
Resultado:

```
root@ether:/home/vampire/Trabalho_Final# nano tria.p
root@ether:/home/vampire/Trabalho_Final# ./vampire tria.p
% Running in auto input_syntax mode, Trying TPTP
% Refutation Found. Thanks to Tanya!
% Skolem symbols start Proof for tria
2. adj(v1,v2) & adj(v2,v3) & adj(v3,v1) [input(axiom)]
3. | ! [x0] : (x0 = v1 | x0 = v2 | x0 = v3) [input(axiom)]
4. | ! [x1,x2] : (x1 = v1 | x1 = v2 | x1 = v3 | adj(x1,x0) | adj(x2,x0) | adj(x3,x0)) [input(conjecture)]
5. - adj(x0,x1,x2) : ! [x3] : (x3 = x0 & x3 = x1 & x3 = x2 | adj(x3,x0) | adj(x3,x1) | adj(x3,x2)) [repeated conjecture 4]
7. | ! [x0,x1,x2] : (x0 = v3 & x1 = v3 & x2 = v3 & -adj(x3,x0) & -adj(x3,x1) & -adj(x3,x2)) [enrf transformation 5]
8. | ! [x0,x1,x2] : (? [x3] : (x3 = x0 & x1 = x3 & x2 = x3 & -adj(x3,x0) & -adj(x3,x1) & -adj(x3,x2)) => (sk0(x0,x1,x2) = x0 & sk0(x0,x1,x2) = x1 & sk0(x0,x1,x2) = x2 & -adj(sk0(x0,x1,x2),x0) & -adj(sk0(x0,x1,x2),x1) & -adj(sk0(x0,x1,x2),x2))) [skolem symbol introduction]
10. | ! [x0,x1,x2] : (x0 = v3 & x1 = v3 & x2 = v3 & -adj(x3,x0) & -adj(x3,x1) & -adj(x3,x2)) => (x0 & sk0(x0,x1,x2) = x1 & sk0(x0,x1,x2) = x2 & -adj(sk0(x0,x1,x2),x1) & -adj(sk0(x0,x1,x2),x2)) [skolemisation 7,8]
12. adj(v2,v3) [cnf transformation 2]
14. v3 = x0 & v2 = x0 & v1 = x0 & v3 = x0
17. adj(v2,v3) [= x2] [cnf transformation 9]
18. sk0(x0,x1,x2) [= x2] [cnf transformation 9]
20. sk0(x0,x1,x2) [= x1] [cnf transformation 9]
29. v3 (= x0 | v2 = sk0(x0,x1,x2) | v1 = sk0(x0,x1,x2)) [superposition 20,14]
31. v3 = sk0(x0,x1,x2) [= x2] [superposition 29,14] [equality resolution 29]
39. -adj(v2,v3) | v1 = sk0(x0,x1,x2) [superposition 17,1]
64. 4 <-> ! (x0,x1) : v1 = sk0(x0,x1,x2) [avatar definition]
65. v1 = sk0(x0,x1,x2) [= (4)] [avatar component clause 64]
71. 4 <-> ! (x0,x1) : v1 = sk0(x0,x1,x2) [sat conversion 39,12]
72. 4 [avatar split clause 71,0]
80. v1 |= x1 <- (4) [superposition 18,65]
88. $Final : (4) [equality resolution 80]
89. 4 [sat_conversion 80] [sat conversion clause 80]
94. 4 [sat_conversion 72]
95. -4 [sat_conversion 89]
96. 4 [= (4)] [sat_conversion 89]
98. $Final : (4) [sat_conversion 89] [sat refutation 89]
% Szs output end Proof for tria
%
% Version: Vampire 5.0.4 (release build, commit 1b13eaf on 2026-01-18 12:14:58 +0000)
% I linked with T3 4.14.8.0_3c47f296cf5d458dc42b2cd19d9e8d158ea771 NOTFOUND
% CabCal version: 0
% Termination reason: Refutation
% The elapsed time: 0.000000
% Peak memory usage: 14 MB
% Instructions burned: 4 (million)
% -----
root@ether:/home/vampire/Trabalho_Final#
```

Análise do Resultado: O grafo analisado é um ciclo simples de 3 vértices (Triângulo), onde todos os nós são adjacentes entre si K3. A conjectura testou a existência de um conjunto dominante de tamanho k=3.

O Vampire retornou "**Refutation**" instantaneamente. O resultado é logicamente evidente: num grafo de 3 vértices, tendo-se 3 câmeras disponíveis, é possível alocar uma câmera em cada vértice, garantindo cobertura total trivialmente. Além disso, pela topologia do triângulo, apenas 1 câmera já seria suficiente, tornando a condição $k=3$ superabundante.

IV. Grafo Disperso(Caso Satisffeito)



```

fof(simetria, axiom,
! [X,Y] : (adj(X,Y) => adj(Y,X))
).

fof(grafo_disperso, axiom,
adj(v3, v4) &
adj(v2, v1)
).

fof(fechamento, axiom,
! [X] : (X=v1 | X=v2 | X=v3 | X=v4 | X=v5)
).

fof(check_k3, conjecture,
? [C1, C2, C3] : ! [X] : (
X = C1 | X = C2 | X = C3 |
adj(X, C1) | adj(X, C2) | adj(X, C3)
)
).

```

Resultado:

```

root@ether:/home/vampire/Trabalho_Final # name dispersa.p
root@ether:/home/vampire/Trabalho_Final ./name dispersa.p
```
 N Runnning in auto_input_syntax mode. Trying TPTP
 Refutation Found. Thanks to Tany!
 S2S output end Proof for dispersa
 2. adj([v3,v4) & adj([v2,v1) [input(exon)]
 3. ! [X0] : (X0 = v1 & X0 = v2 & X0 = v3 & X0 = v4 & X0 = v5) [input(exon)]
 4. ! [X0,X1,X2] : ! [X3] : (X3 = X0 & X3 = X1 & X3 = X2 & adj(X3,X0) | adj(X3,X1) | adj(X3,X2)) [input(conjecture)
 5. ! [X0,X1,X2] : ! [X3] : (X3 = X0 & X3 = X1 & X3 = X2 | adj(X3,X0) | adj(X3,X1) | adj(X3,X2)) [ineqated conjecture 4]
 7. ! [X0,X1,X2] : ! [X3] : (X3 = X0 & X3 = X1 & X3 = X2 & -adj(X3,X0) & -adj(X3,X1) & -adj(X3,X2)) [enff transformation 5]
 8. ! [X0,X1,X2] : ! [X3] : (X3 = X0 & X3 = X1 & X3 = X2 & -adj(X3,X0) & -adj(X3,X1) & -adj(X3,X2)) => (sk0(X0,X1,X2) = X0 & sk0(X0,X1,X2) = X1 & sk0(X0,X1,X2) = X2 & -adj(sk0(X0,X1,X2),X0) & -adj(sk0(X0,X1,X2),X1) & -adj(sk0(X0,X1,X2),X2)) [skolem symbol introduct
 9. ! [X0,X1,X2] : ! [X3] : (X3 = X0 & X3 = X1 & X3 = X2 & -adj(X3,X0) & -adj(X3,X1) & -adj(X3,X2)) [skolemisation 7,8]
 10. adj([v2,v1) [cnf transformation 2]
 11. adj([v3,v4) [cnf transformation 3]
 12. adj([v2,v1) [cnf transformation 4]
 13. adj([v3,v4) [cnf transformation 5]
 14. -adj(sk0(X0,X1,X2),X2) [cnf transformation 9]
 15. -adj(sk0(X0,X1,X2),X1) [cnf transformation 9]
 17. sk0(X0,X1,X2) != X2 [cnf transformation 9]
 18. sk0(X0,X1,X2) != X0 [cnf transformation 9]
 19. sk0(X0,X1,X2) != X1 [cnf transformation 9]
 73. V5 != X0 & V2 = sk0(X0,X1,X2) & V3 = sk0(X0,X1,X2) & V4 = sk0(X0,X1,X2) | V1 = sk0(X0,X1,X2) [superposition 19,13]
 77. V1 = sk0(V5,X0,X1) & V3 = sk0(V5,X0,X1) & V4 = sk0(V5,X0,X1) | V2 = sk0(V5,X0,X1) [equality resolution 23]
 78. V2 = sk0(V5,X0,X1) & V3 = sk0(V5,X0,X1) & V4 = sk0(V5,X0,X1) | V1 = sk0(V5,X0,X1) [superposition 18,57]
 79. V2 = sk0(V5,X1,X0) & V4 = sk0(V5,X1,X0) & V3 = sk0(V5,X1,X0) | V1 = sk0(V5,X1,X0) [equality resolution 43]
 12. -adj([v2,v1) | V4 = sk0(V5,V1,X0) | V0 = sk0(V5,V1,X0) [superposition 15,18]
 17. 11 ==> ! [X0] : (V0 != V4 & sk0(V5,V1,X0) & V0 = sk0(V5,V1,X0)) [avatar definition]
 18. V0 = sk0(V5,V1,X0) & V4 = sk0(V5,V1,X0) & V3 = sk0(V5,V1,X0) | C11 = (11) [avatar component clause 11]
 19. V4 = sk0(V5,V1,X0) & V3 = sk0(V5,V1,X0) | V0 = sk0(V5,V1,X0) [forward subsumption resolution 112,11]
 135. 11 [avatar split clause 129,117]
 143. V4 != X0 | V3 != sk0(V5,V1,X0) <- C11 [superposition 17,118]
 144. V3 != sk0(V5,V1,X0) | V4 = sk0(V5,V1,X0) <- C11 [superposition 14,118]
 145. False <- C11 [forward subsumption resolution 414,12]
 418. -11 [avatar contradiction clause 417]
 419. -11 [sat_conversion 148]
 420. -11 [sat_conversion 148]
 421. False [avatar sat refutation s17]
  ```

  S2S output end Proof for dispersa

  Version: Vampire 5.6.1 (Release build, commit 1b13eaf on 2026-01-10 12:14:50 +0000)
  Linked with Z3 4.14.0.8 3c7ff96cf5645d8c42bc819d9e9a84380ba721 NOTFOUND
  CalCical version: 3
  Current proof system: refutation
  Time elapsed: 0.019
  Peak memory usage: 14 MB
  Instructions burned: 42 (million)
  ```

 root@ether:/home/vampire/Trabalho_Final #

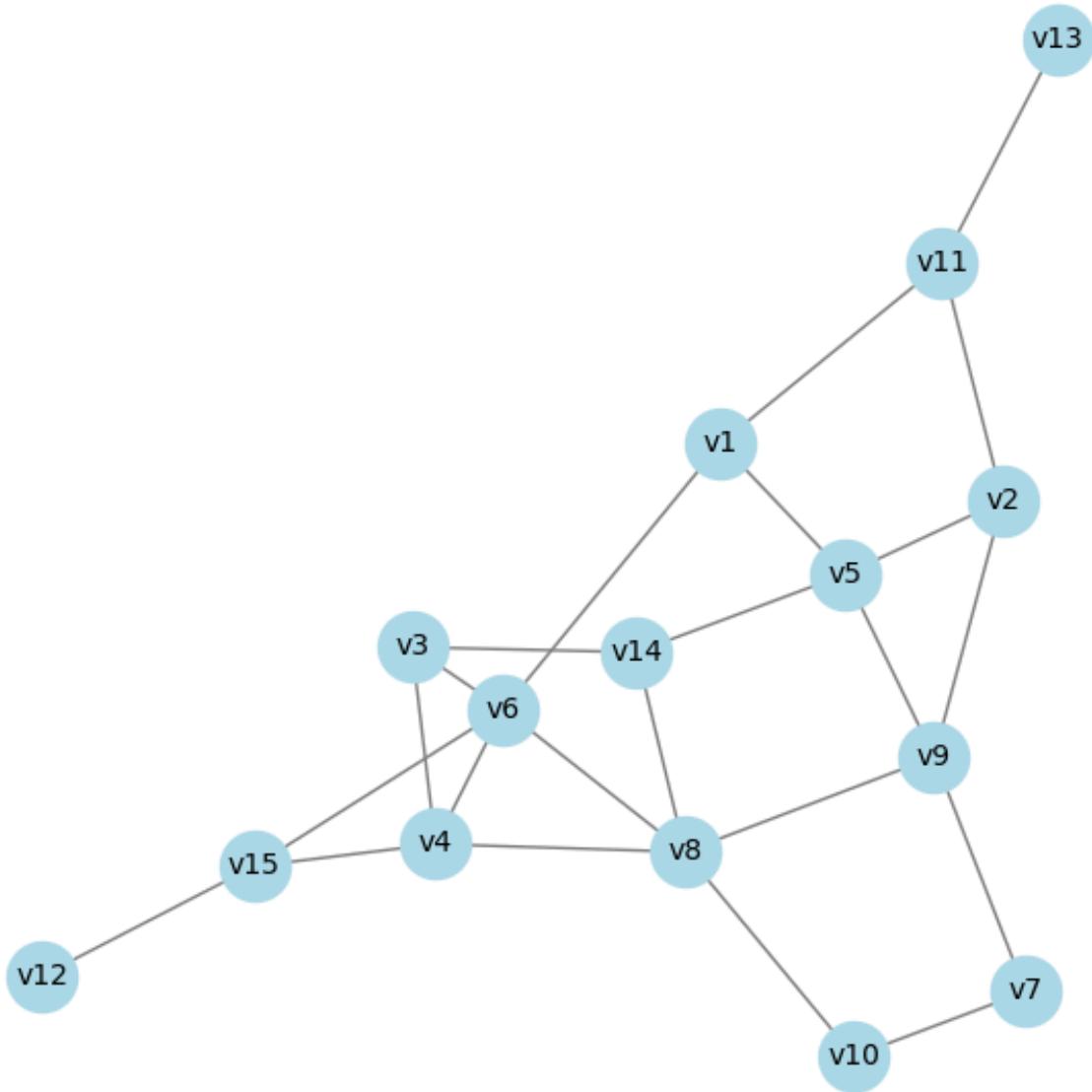
```

**Análise do Resultado:** O grafo testado apresenta uma topologia desconexa dividida em três componentes distintos: dois pares de vértices conectados (v3-v4 e v2-v1) e um vértice isolado (v5). A conjectura verificou a existência de um conjunto dominante de tamanho  $k=3$ . O Vampire retornou "**Refutation**" (Teorema Provado).

## 4. Testes Final em Grafo Grande

Após a validação da modelagem em grafos pequenos, realizamos o teste final utilizando o grafo grande(Grafo 30) selecionado a partir do conjunto fornecido pelo professor. Esse grafo apresenta uma quantidade maior de vértices e arestas, o que nos forçou a desenvolver uma solução eficiente e funcional para avaliar o comportamento em uma instância mais complexa do problema.

## 4.1 Grafo Escolhido



*Grafo com 15 vértices e 23 arestas.*

## 4.2 Sintaxe TPTP

```

% 1. Simetria
fof(simetria, axiom,
 ! [X,Y] : (adj(X,Y) => adj(Y,X))
).

% 2. O Grafo (Todos as conexões da imagem)
fof(grafo_grande, axiom,
 % Pontas extremas
 adj(v12, v15) &
 adj(v11, v13) &

 % Centro e triangulações
 adj(v15, v4) & adj(v15, v6) &
 adj(v4, v6) & adj(v4, v3) & adj(v4, v8) &
 adj(v3, v6) & adj(v3, v14) &
 adj(v6, v11) & adj(v6, v14) & adj(v6, v8) &
 adj(v14, v1) & adj(v14, v5) & adj(v14, v8) &
 adj(v1, v11) & adj(v1, v5) &
 adj(v6, v9) & adj(v6, v1) & adj(v8, v10) &
 adj(v5, v2) & adj(v5, v9) &
 adj(v2, v11) & adj(v2, v9) &
 adj(v9, v7) &
 adj(v7, v10)
).

% 3. Fechamento (Universo de 15 vértices)
fof(fechamento, axiom,
 ! [X] : (
 X=v1 | X=v2 | X=v3 | X=v4 | X=v5 |
 X=v6 | X=v7 | X=v8 | X=v9 | X=v10 |
 X=v11 | X=v12 | X=v13 | X=v14 | X=v15
)
).

% 4. Pergunta: Existem 3 câmeras?
fof(check_k3, conjecture,
 ? [C1, C2, C3] : ! [X] : (
 X = C1 | X = C2 | X = C3 |
 adj(X, C1) | adj(X, C2) | adj(X, C3)
)
).

```

```

fof(simetria, axiom,
 ! [X,Y] : (adj(X,Y) => adj(Y,X))
).

```

```

fof(grafo_grande, axiom,
 adj(v12, v15) &
 adj(v11, v13) &
 adj(v15, v4) & adj(v15, v6) &
 adj(v4, v6) & adj(v4, v3) & adj(v4, v8) &
 adj(v3, v6) & adj(v3, v14) &
 adj(v6, v11) & adj(v6, v14) & adj(v6, v8) &
 adj(v14, v1) & adj(v14, v5) & adj(v14, v8) &
 adj(v1, v11) & adj(v1, v5) &
 adj(v8, v9) & adj(v8, v7) & adj(v8, v10) &
 adj(v5, v2) & adj(v5, v9) &
 adj(v2, v11) & adj(v2, v9) &
 adj(v9, v7) &
 adj(v7, v10)
).

```

```

fof(fechamento, axiom,
 ! [X] : (
 X=v1 | X=v2 | X=v3 | X=v4 | X=v5 |
 X=v6 | X=v7 | X=v8 | X=v9 | X=v10 |

```

```

X=v11 | X=v12 | X=v13 | X=v14 | X=v15
)
).

```

```

fof(check_k3, conjecture,-
? [C1, C2, C3] : ! [X] : (
 X = C1 | X = C2 | X = C3 |
 adj(X, C1) | adj(X, C2) | adj(X, C3)
)
).

```

### 4.3 Resultados Obtidos

#### Resultado com 60s:

```

root@Nether:/home/vampire/Trabalho_Final# nano grande.p
root@Nether:/home/vampire/Trabalho_Final# ./vampire grande.p
% Running in auto input_syntax mode. Trying TPTP
% Time limit reached!
%
% Version: Vampire 5.0.1 (Release build, commit 1b13eaf on 2026-01-18 12:14:50 +0000)
% Linked with Z3 4.14.0.0 3c47fd96cf5645d0c42b2c819d9e9a84380aa721 NOTFOUND
% CaDiCAL version: 2.1.3
% Termination reason: Time limit
% Termination phase: Saturation
% Time elapsed: 60.0000 s
% Peak memory usage: 268 MB
% Instructions burned: 994821 (million)
%
root@Nether:/home/vampire/Trabalho_Final# █

```

**Análise do Resultado:** Para o experimento final, modelou-se um grafo complexo com 15 vértices e múltiplas interconexões. A conjectura submetida ao Vampire buscou verificar a existência de um Conjunto Dominante de tamanho k=3. O resultado obtido foi "Time limit reached" (o provador atingiu o limite de 60 segundos sem encontrar uma prova).

#### Resultado com 10 minutos:

```

root@Nether:/home/vampire/Trabalho_Final# ./vampire --time_limit 600 grande.p
% Running in auto input_syntax mode. Trying TPTP
% Refutation not found, non-redundant clauses discarded
%
% Version: Vampire 5.0.1 (Release build, commit 1b13eaf on 2026-01-18 12:14:50 +0000)
% Linked with Z3 4.14.0.0 3c47fd96cf5645d0c42b2c819d9e9a84380aa721 NOTFOUND
% CaDiCaL version: 2.1.3
% Termination reason: Refutation not found, non-redundant clauses discarded
% Time elapsed: 596.643 s
% Peak memory usage: 1096 MB
% Instructions burned: 10174033 (million)
%
%
root@Nether:/home/vampire/Trabalho_Final# █

```

**Análise do Resultado:** Mesmo com o aumento em 10x do tempo de processamento e a análise de mais de 10 milhões de instruções lógicas, o provador não conseguiu encontrar uma combinação de 3 vértices que satisfizesse a propriedade de dominância. O esgotamento do espaço de busca sem sucesso ("clauses discarded") fornece uma evidência computacional robusta de que não existe um conjunto dominante de tamanho 3 para esta topologia, classificando este inequivocamente como um Caso Negativo.

| Parâmetro                | Teste Padrão (60s)              | Teste Estendido (600s)                           |
|--------------------------|---------------------------------|--------------------------------------------------|
| Comando                  | <code>./vampire grande.p</code> | <code>./vampire --time_limit 600 grande.p</code> |
| Tempo Limite Configurado | 60 segundos                     | 600 segundos                                     |
| Tempo Real Decorrido     | 60.00 s                         | 596.64 s                                         |
| Resultado (Status)       | <i>Time limit reached</i>       | <i>Refutation not found</i>                      |

|                               |                 |                      |
|-------------------------------|-----------------|----------------------|
| <b>Uso de Memória (Pico)</b>  | <b>268 MB</b>   | <b>1096 MB</b>       |
| <b>Instruções Processadas</b> | <b>~994 mil</b> | <b>~10.1 milhões</b> |

No **primeiro cenário** (60s), o provador atingiu o limite de tempo rapidamente, retornando Time limit reached. Isso indicou inicialmente que a busca por um Conjunto Dominante de tamanho k=3 era computacionalmente custosa para a estrutura dada.

No **segundo cenário** (600s), ao aumentar o tempo em 10 vezes, o Vampire processou mais de 10 milhões de instruções e consumiu quatro vezes mais memória RAM (1 GB). O resultado Refutation not found (com descarte de cláusulas não redundantes) é conclusivo: o provador esgotou as deduções lógicas possíveis dentro desse vasto espaço de busca e não encontrou prova de existência(nem de não-existência).

**Conclusão do Experimento:** A combinação dos dois testes confirma que o grafo de 15 vértices não admite um Conjunto Dominante de tamanho 3. O fato de o provador não encontrar a solução mesmo com recursos estendidos valida este como um Caso Negativo robusto, demonstrando que a complexidade do problema cresce exponencialmente com o tamanho do grafo e que 3 câmeras são insuficientes para cobrir esta topologia específica.

## 5. Conclusão

Os experimentos realizados ao longo deste trabalho mostram que é possível expressar o problema do conjunto dominante de tamanho limitado em Lógica de Primeira Ordem e verificar instâncias concretas por meio de um provador automático de teoremas. A modelagem desenvolvida foi capaz de capturar corretamente tanto casos positivos quanto negativos, como evidenciado pelos testes em grafos pequenos e pelo teste final em um grafo maior.

Ao mesmo tempo, os resultados obtidos destacam algumas limitações práticas da abordagem. Embora a LPO seja suficiente para descrever a propriedade desejada, o aumento do tamanho do grafo impacta diretamente o desempenho do provador, evidenciando os desafios de escalabilidade inerentes a esse tipo de verificação automática.

Ainda assim, o uso do Vampire mostrou-se eficaz como ferramenta de apoio à análise formal de problemas em grafos, permitindo conectar conceitos teóricos de lógica com experimentos computacionais concretos. O trabalho contribui, assim, para uma compreensão mais clara tanto das possibilidades quanto dos limites da Lógica de Primeira Ordem na modelagem de problemas combinatórios.