UNIVERSITÀ DEGLI STUDI DI SALERNO

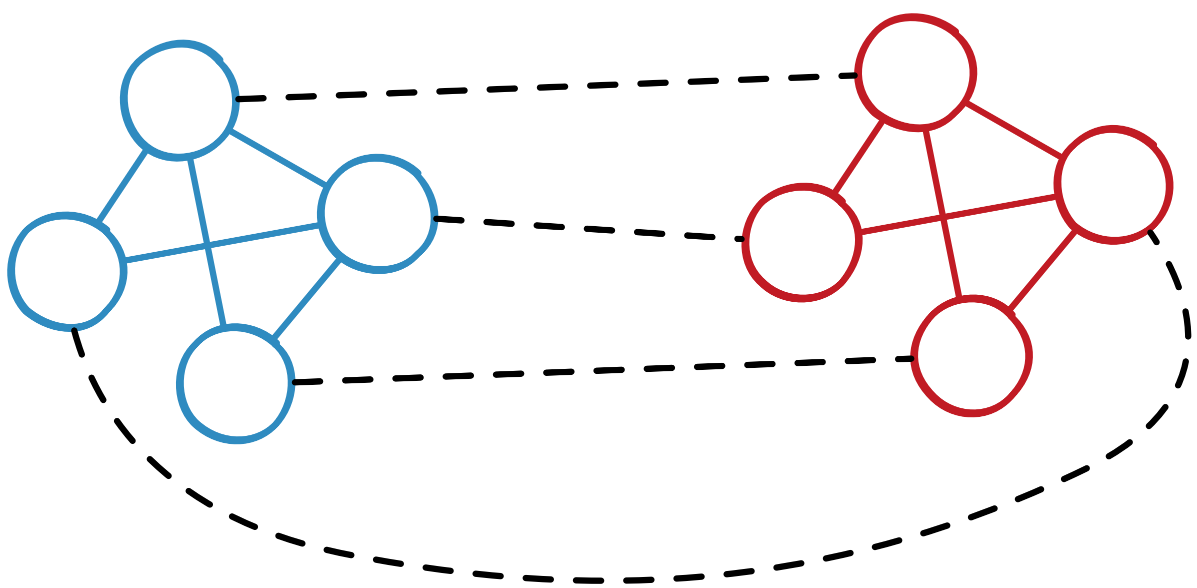
CHALLENGING MEMORY AND TIME COMPLEXITY OF SUBGRAPH ISOMORPHISM PROBLEM WITH VF3

MIVIA

OVERVIEW

GRAPH MATCHING ALGORITMS

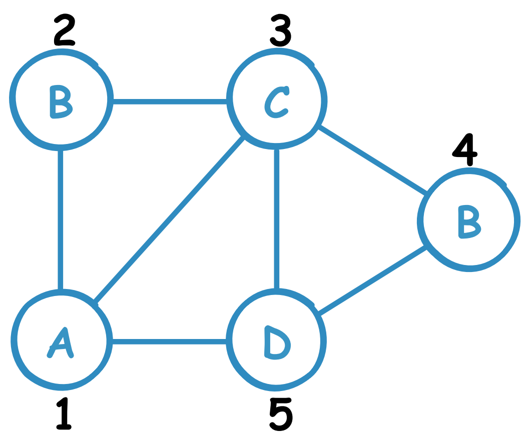
- Exact Graph Matching
  - Structure Preserving Mapping
    - Adjacency-preserving
    - Non-adjacency preserving
  - NP-Complete Problem



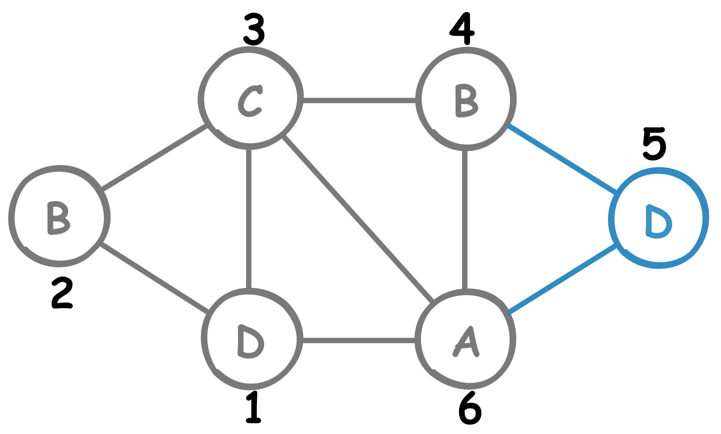
SIMILARITY BY SUBGRAPH ISOMORPHISM

WHAT IS SUBGRAPH ISOMORPHISM?

- Given a graph:
  - Is it inside another graph?
  - How many times?
  - Where?



- Common applications:
  - Pattern search or Graph querying
  - Graph learning algorithms
  - Graph clustering



WHERE DID WE BEGIN FROM? VF2!

- **A (sub) graph isomorphism algorithm for matching large graphs** - IEEE Transactions on PAMI - L. Cordella, P. Foggia, C. Sansone, M. Vento – 2004.

- **VF2 is currently used in**:
  - Boost C++ library
  - Networkx python library
  - MIT Courses

VF3: IN BRIEF

Inherited

- State Space Representation
  - Each state is partial solution
  - A goal is consistent complete solution
- Depth-First search with backtracking
  - State space as a tree by a total order relationship
- Feasibility rules to explore the space
  - Consistent states only
  - 2-levels Look-ahead

VF3: IN BRIEF

Inherited

- State Space Representation
  - Each state is partial solution
  - A goal is consistent complete solution
- Depth-First search with backtracking
  - State space as a tree by a total order relationship
- Feasibility rules to explore the space
  - Consistent states only
  - 2-levels Look-ahead

New

- Pattern Pre-processing
  - The first graph is explored before starting.
- Straightened look-ahead
  - Node classification
    - Structural and Semantic features
- New state transition function
  - Significantly reduced the set where the next pairs are searched.

VF3: IN BRIEF

Inherited

- State Space Representation
  - Each state is partial solution
  - A goal is consistent complete solution
- Depth-First search with backtracking
  - State space as a tree by a total order relationship
- Feasibility rules to explore the space
  - Consistent states only
  - 2-levels Look-ahead

Unchanged Complexity

- Linear in space
- Quadratic in time (avg)

New

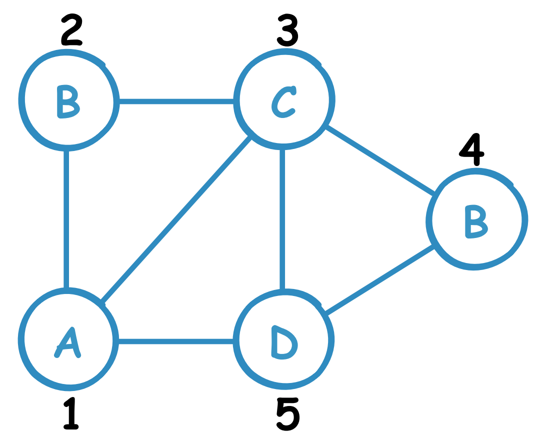
- Pattern Pre-processing
  - The first graph is explored before starting.
- Straightened look-ahead
  - Node classification
    - Structural and Semantic features
- New state transition function
  - Significantly reduced the set where the next pairs are searched.

But faster!

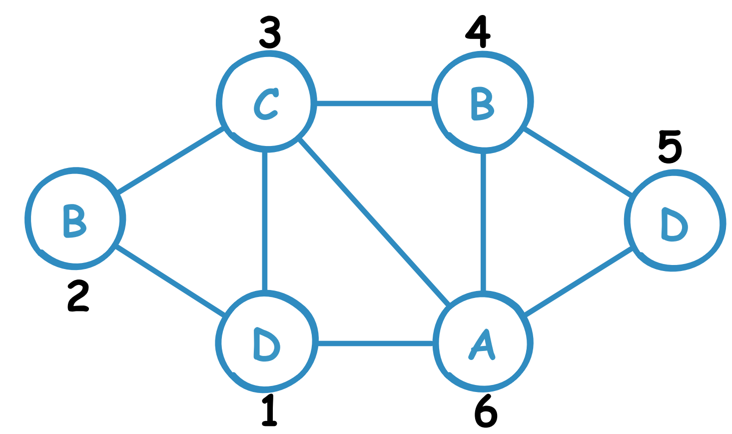
(From 10 to 1000 times)

VF3: HOW DOES IT WORK? (WARMUP)

1. Classify the nodes
   - Structural features
   - Semantic features
2. Order the nodes
   - Most constrained first
   - Most rare first
   - Generate a node sequence
3. Pre-process the pattern
   - Prepare the structures
   - Generate a coverage tree



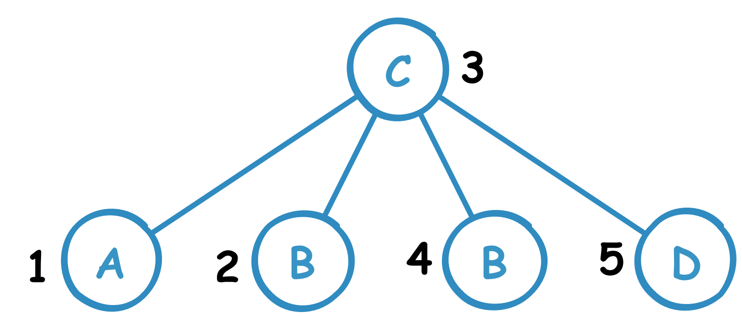
Pattern (G_1)



Target (G_2)

Class	1	2	3	4
Label	D	B	C	A

$N = \{3, 1, 5, 2, 4\}$



VF3: HOW DOES IT WORK? (EXPLORATION)

4. Exploring the state space
   - Start from an empty solution
   - Generate new states
     - Iteratively add a new matching couple by following the exploration sequence
     - Check for the consistence
   - Move through consistent states

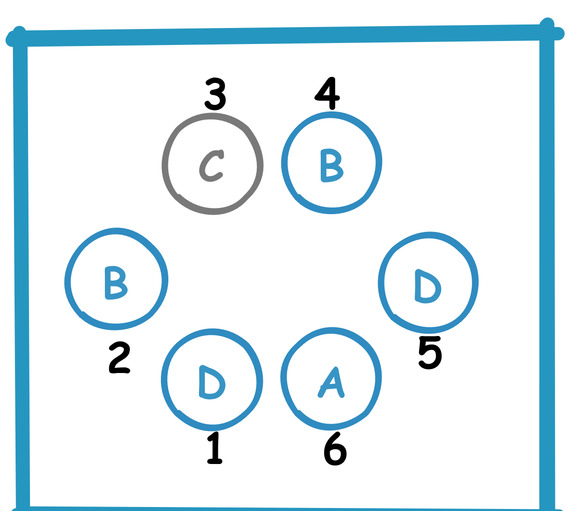
State

S_0

Mapping

$M(S_0)=\emptyset$

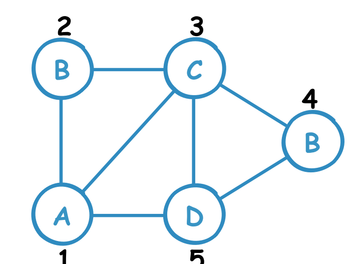
$N = \{3, 1, 5, 2, 4\}$



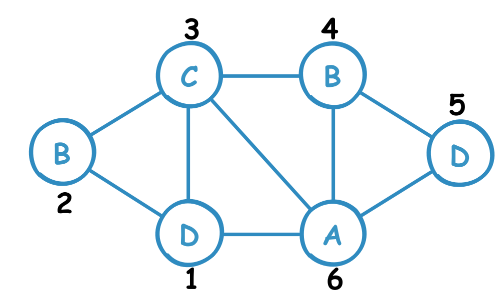
Candidates of G_2

$G_1(S_0)$

$G_2(S_0)$



G_1



G_2

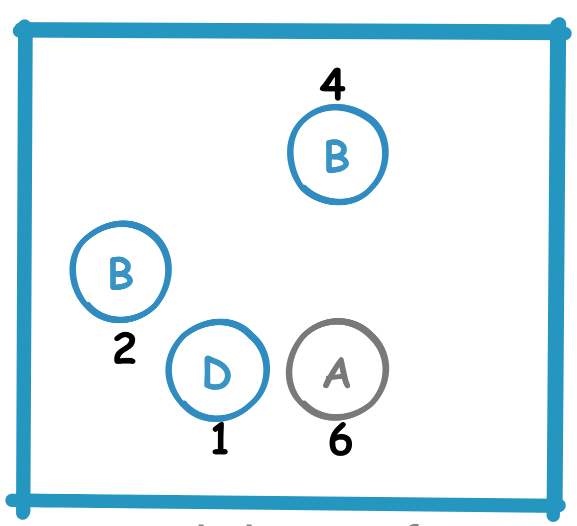
VF3: HOW DOES IT WORK? (EXPLORATION)

4. Exploring the state space

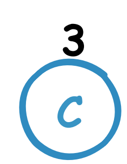
- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$

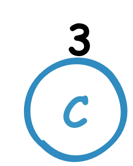
N = {3, 1, 5, 2, 4}



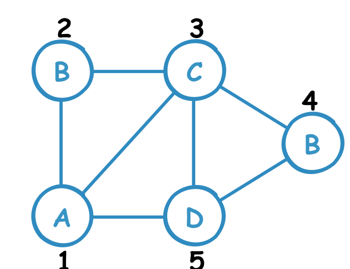
Candidates of G_2



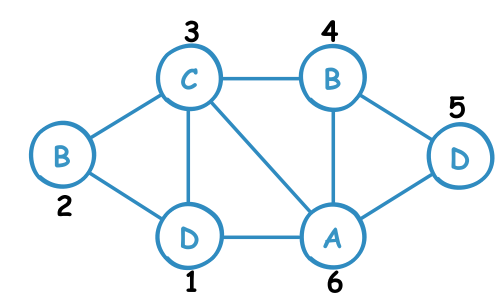
$G_1(S_1)$



$G_2(S_1)$



G_1



G_2

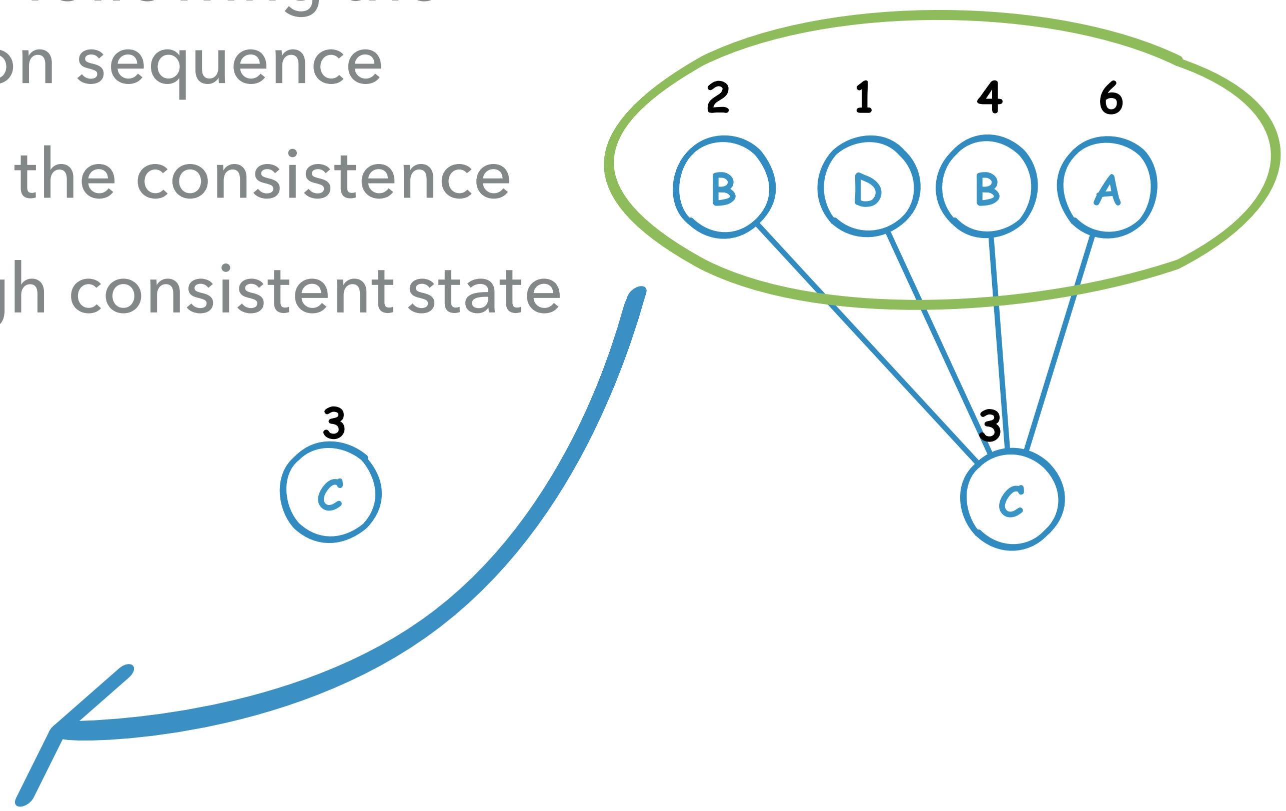
VF3: HOW DOES IT WORK? (EXPLORATION)

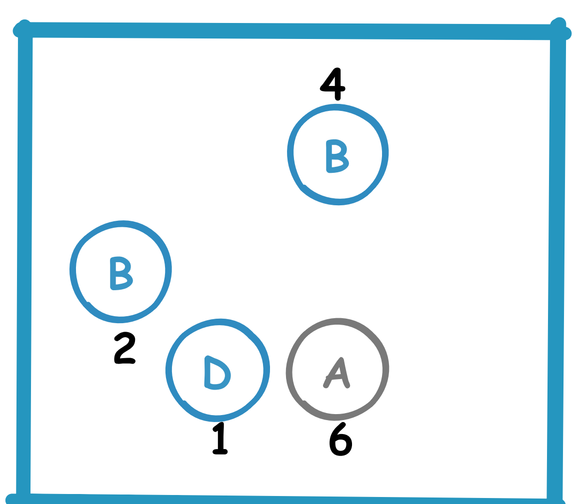
4. Exploring the state space

- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$

N = {3, 1, 5, 2, 4}

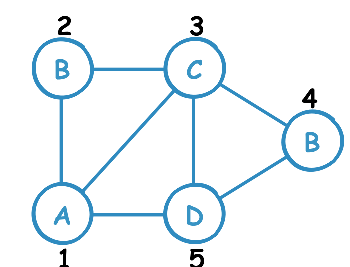




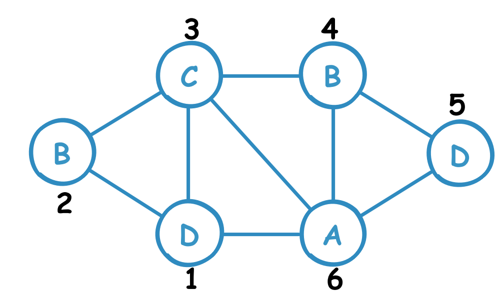
Candidates of G_2

$G_1(S_1)$

$G_2(S_1)$



G_1



G_2

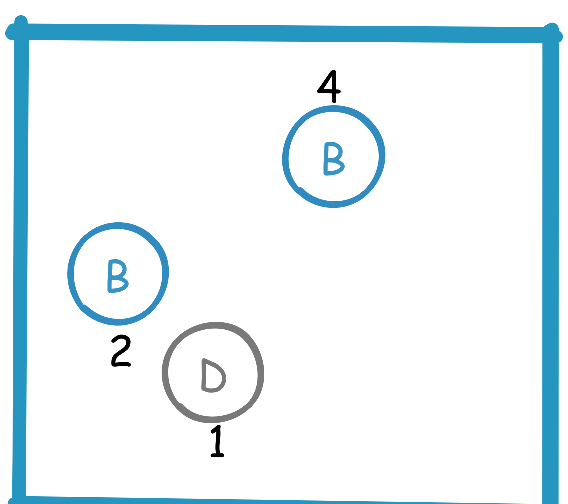
VF3: HOW DOES IT WORK? (EXPLORATION)

4. Exploring the state space

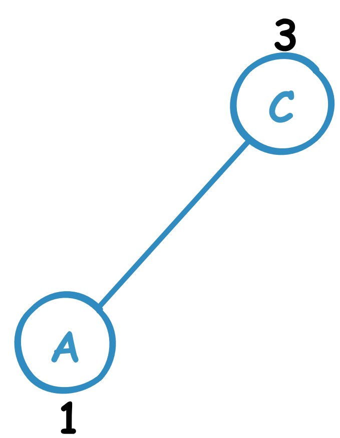
- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$
S_2	$M(S_2) = \{(3, 3), (1, 6)\}$

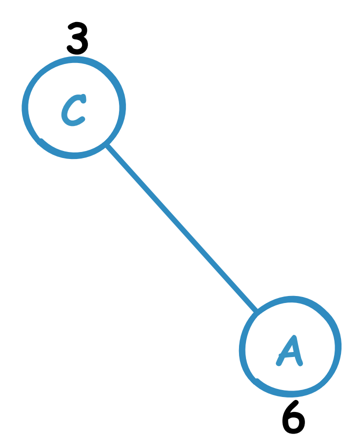
N = {3, 1, 5, 2, 4}



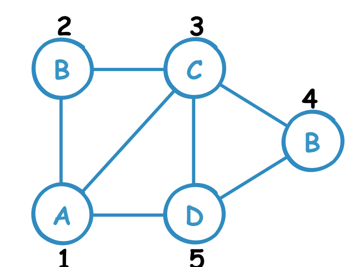
Candidates of G_2



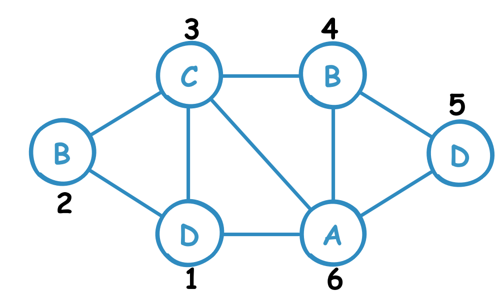
$G_1(S_2)$



$G_2(S_2)$



G_1



G_2

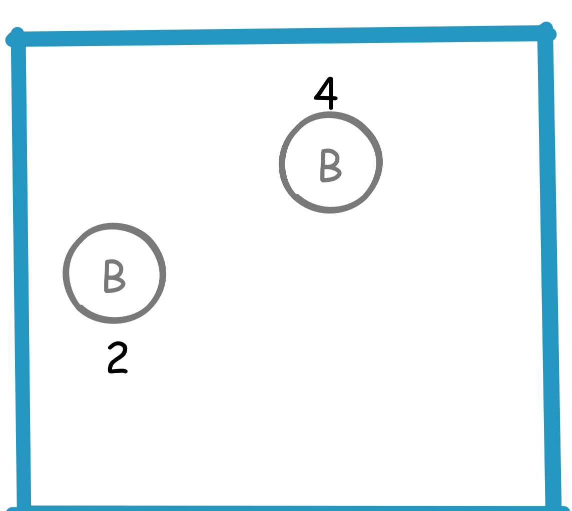
VF3: HOW DOES IT WORK? (EXPLORATION)

4. Exploring the state space

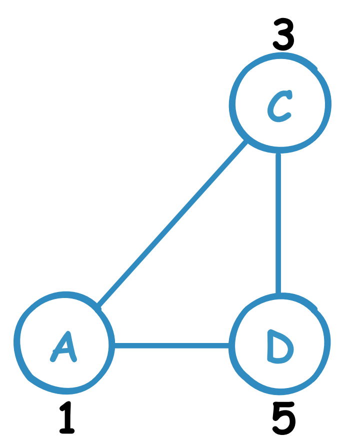
- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

State	Mapping
S_0	$M(S_0)=\varnothing$
S_1	$M(S_1) = \{(3, 3)\}$
S_2	$M(S_2) = \{(3, 3), (1, 6)\}$
S_3	$M(S_3) = \{(3, 3), (1, 6), (5, 1)\}$

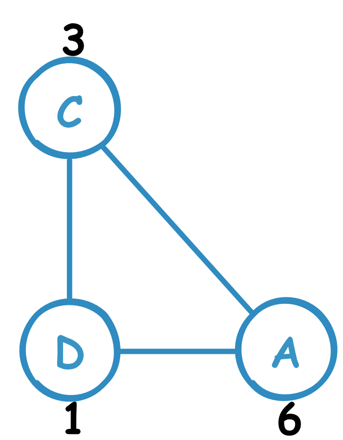
N = {3, 1, 5, 2, 4}



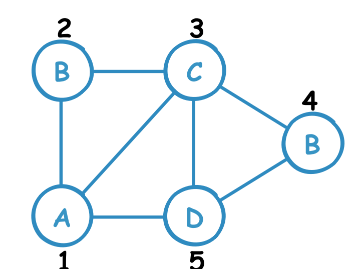
Candidates of G_2



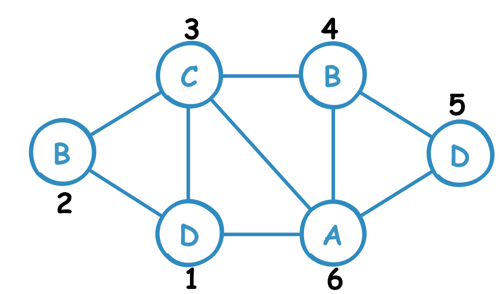
$G_1(S_3)$



$G_2(S_3)$



G_1



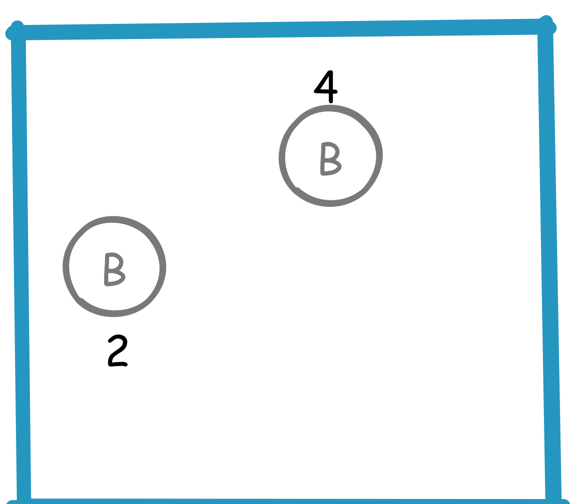
G_2

VF3: HOW DOES IT WORK? (EXPLORATION)

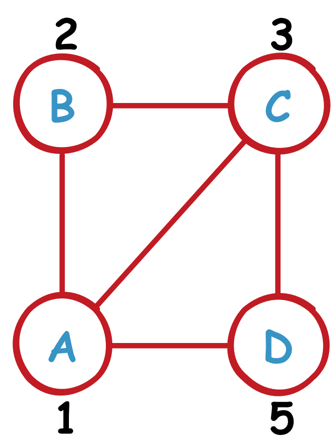
4. Exploring the state space
   - Start from an empty solution
   - Generate new states
     - Iteratively add a new matching couple by following the exploration sequence
     - Check for the consistence
   - Move through consistent state

State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$
S_2	$M(S_2) = \{(3, 3), (1, 6)\}$
S_3	$M(S_3) = \{(3, 3), (1, 6), (5, 1)\}$
S_4	$\cancel{M(S_4) = \{(3, 3), (1, 6), (5, 1), (2, 2)\}}$

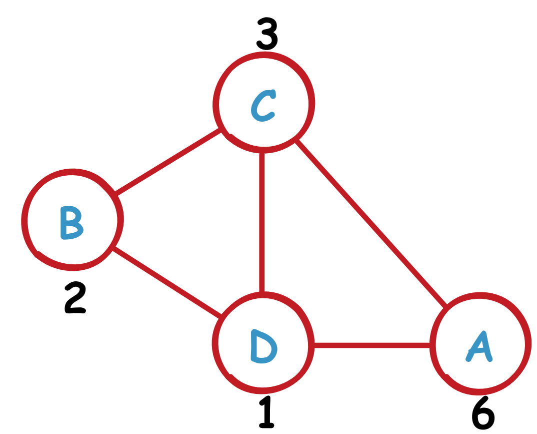
N = {3, 1, 5, 2, 4}



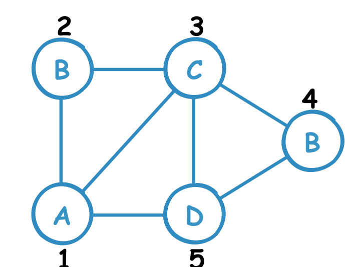
Candidates of G_2



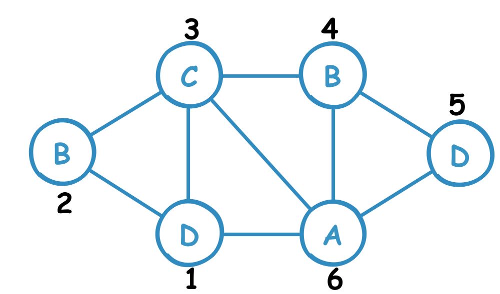
$G_1(S_4)$



$G_2(S_4)$



G_1



G_2

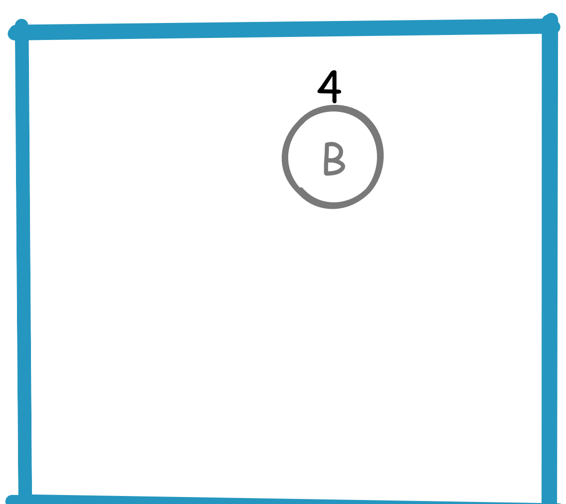
VF3: HOW DOES IT WORK? (EXPLORATION)

4. Exploring the state space

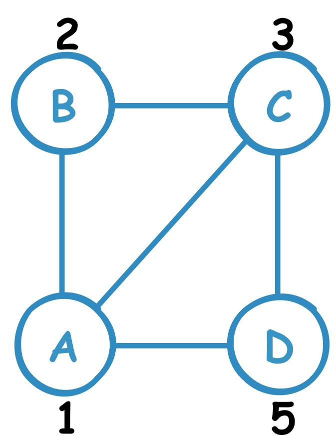
- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$
S_2	$M(S_2) = \{(3, 3), (1, 6)\}$
S_3	$M(S_3) = \{(3, 3), (1, 6), (5, 1)\}$
S_4	$\cancel{M(S_4) = \{(3, 3), (1, 6), (5, 1), (2, 2)\}}$
S_5	$M(S_5) = \{(3, 3), (1, 6), (5, 1), (2, 4)\}$

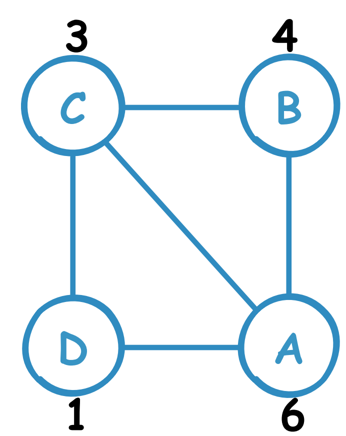
$N = \{3, 1, 5, 2, 4\}$



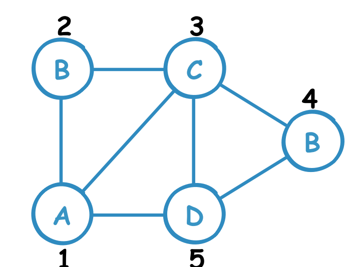
Candidates of G_2



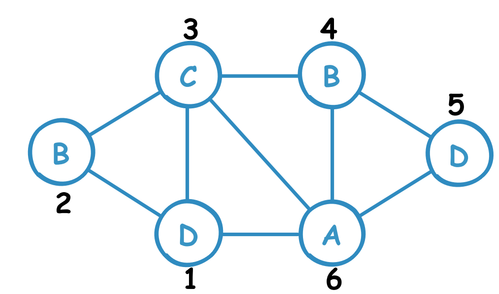
$G_1(S_5)$



$G_2(S_5)$



G_1



G_2

VF3: HOW DOES IT WORK? (EXPLORATION)

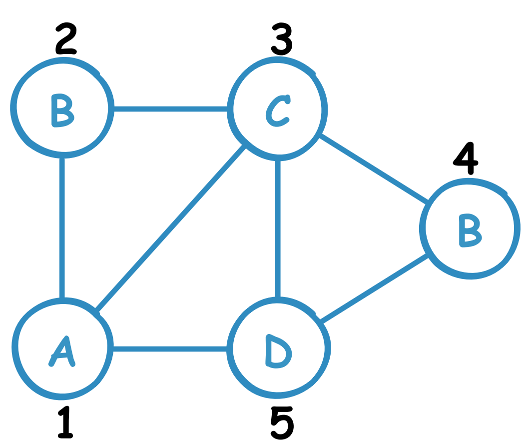
4. Exploring the state space

- Start from an empty solution
- Generate new states
  - Iteratively add a new matching couple by following the exploration sequence
  - Check for the consistence
- Move through consistent state

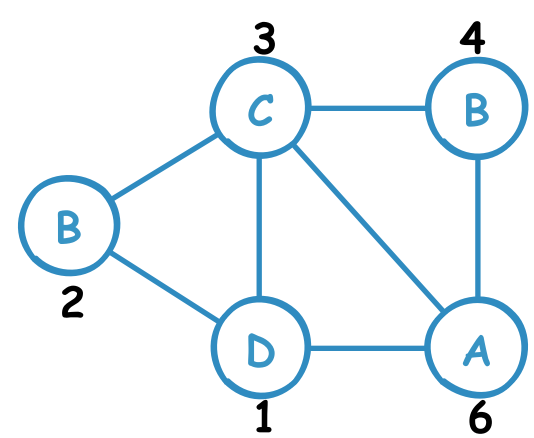
State	Mapping
S_0	$M(S_0)=\emptyset$
S_1	$M(S_1) = \{(3, 3)\}$
S_2	$M(S_2) = \{(3, 3), (1, 6)\}$
S_3	$M(S_3) = \{(3, 3), (1, 6), (5, 1)\}$
S_4	~~$M(S_4) = \{(3, 3), (1, 6), (5, 1), (2, 2)\}$~~
S_5	$M(S_5) = \{(3, 3), (1, 6), (5, 1), (2, 4)\}$
S_6	$M(S_6) = \{(3, 3), (1, 6), (5, 1), (2, 4), (4, 2)\}$

N = {3, 1, 5, 2, 4}

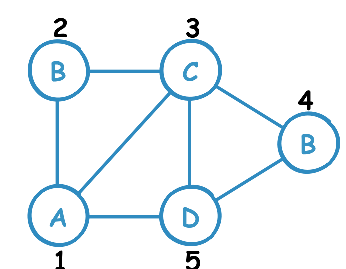
Candidates of G_2



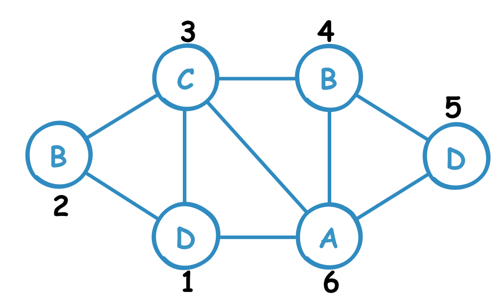
$G_1(S_6)$



$G_2(S_6)$



G_1



G_2

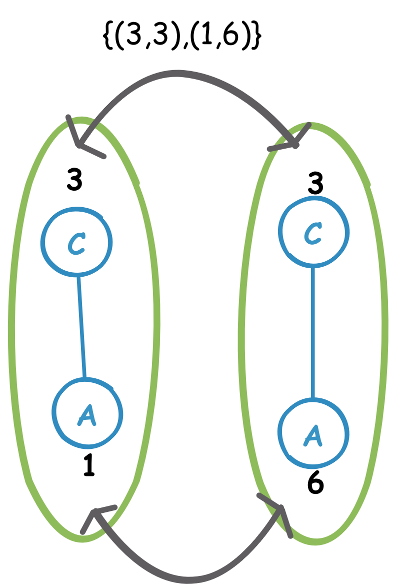
VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency? → Feasibility Rules

VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency? → Feasibility Rules

Core Rule

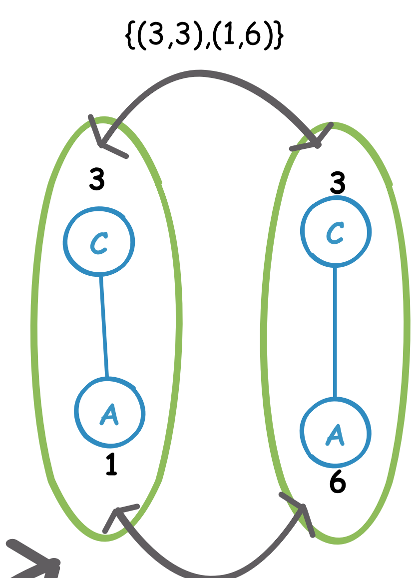


MIVIA

VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency? → Feasibility Rules

Core Rule



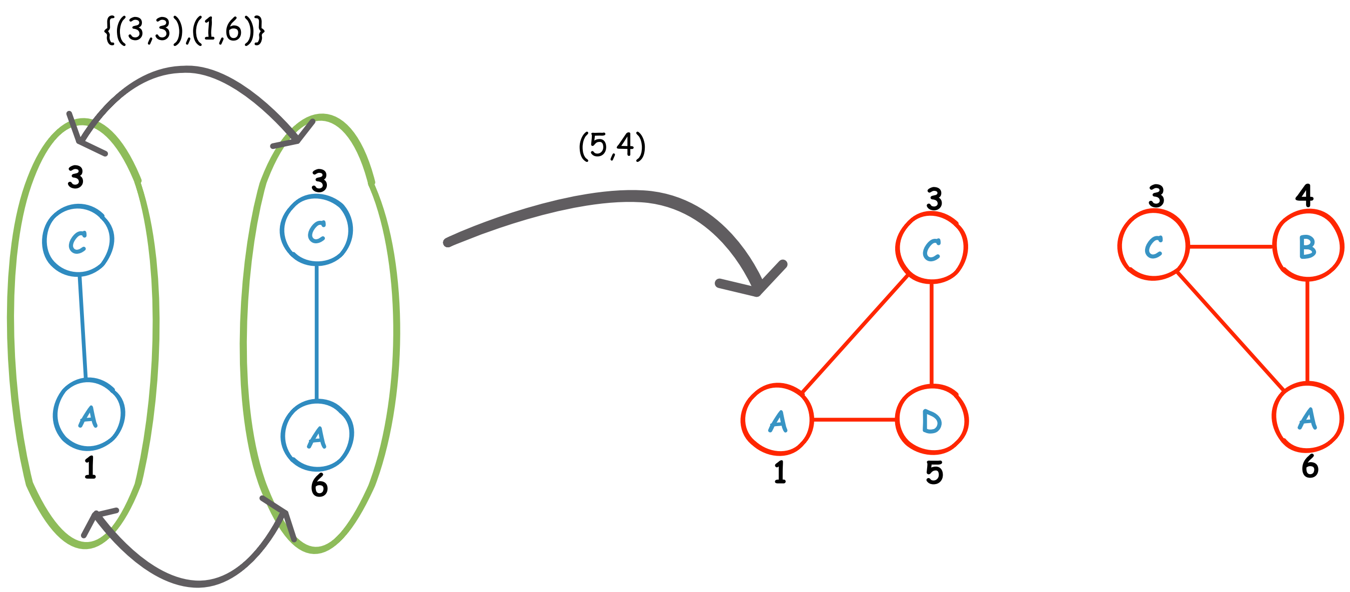
Core Sets

VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency?

Feasibility Rules

Core Rule

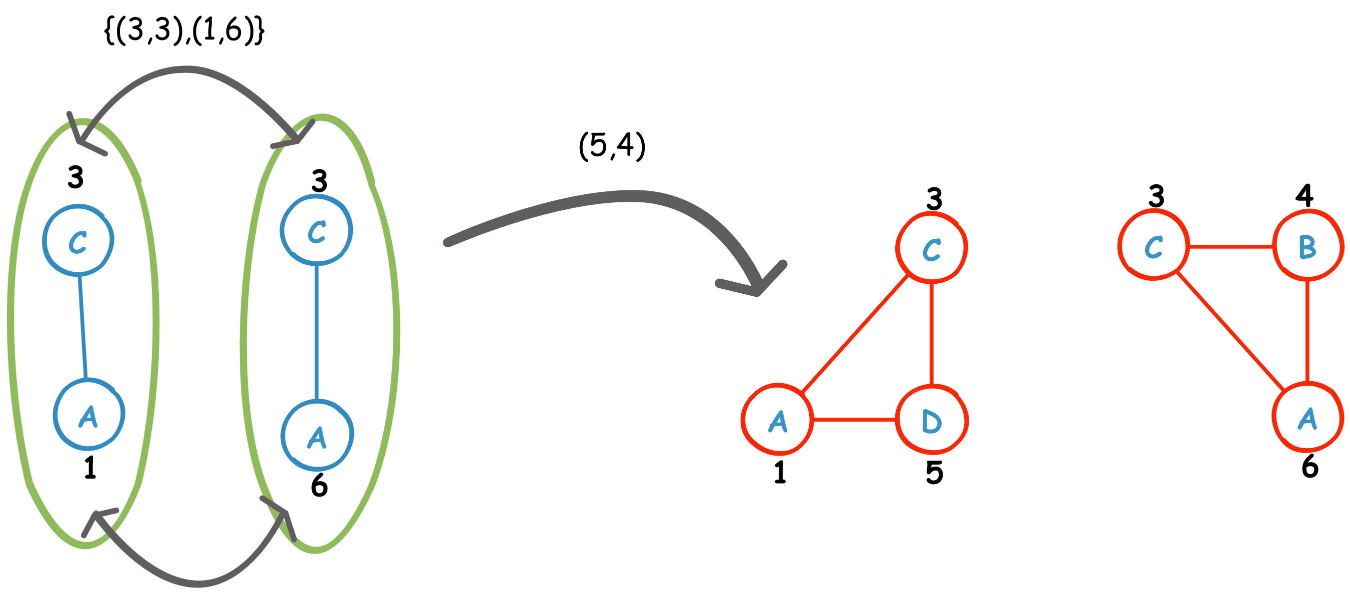


Semantic Inconsistence

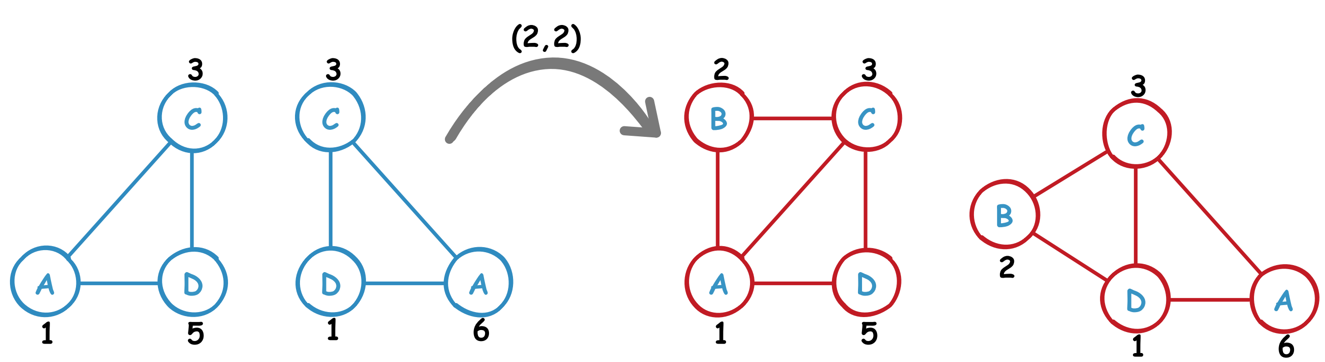
VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency? → Feasibility Rules

Core Rule



Semantic Inconsistence



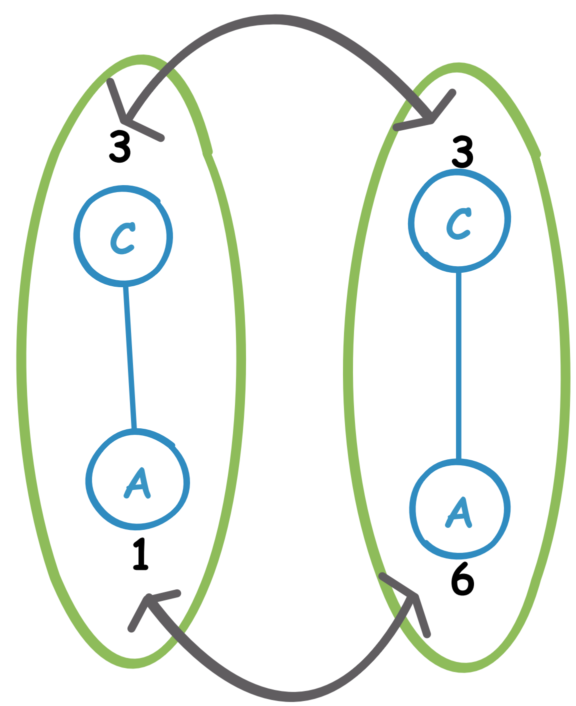
Structural Inconsistence

VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency?

Feasibility Rules

Look-ahead Rules

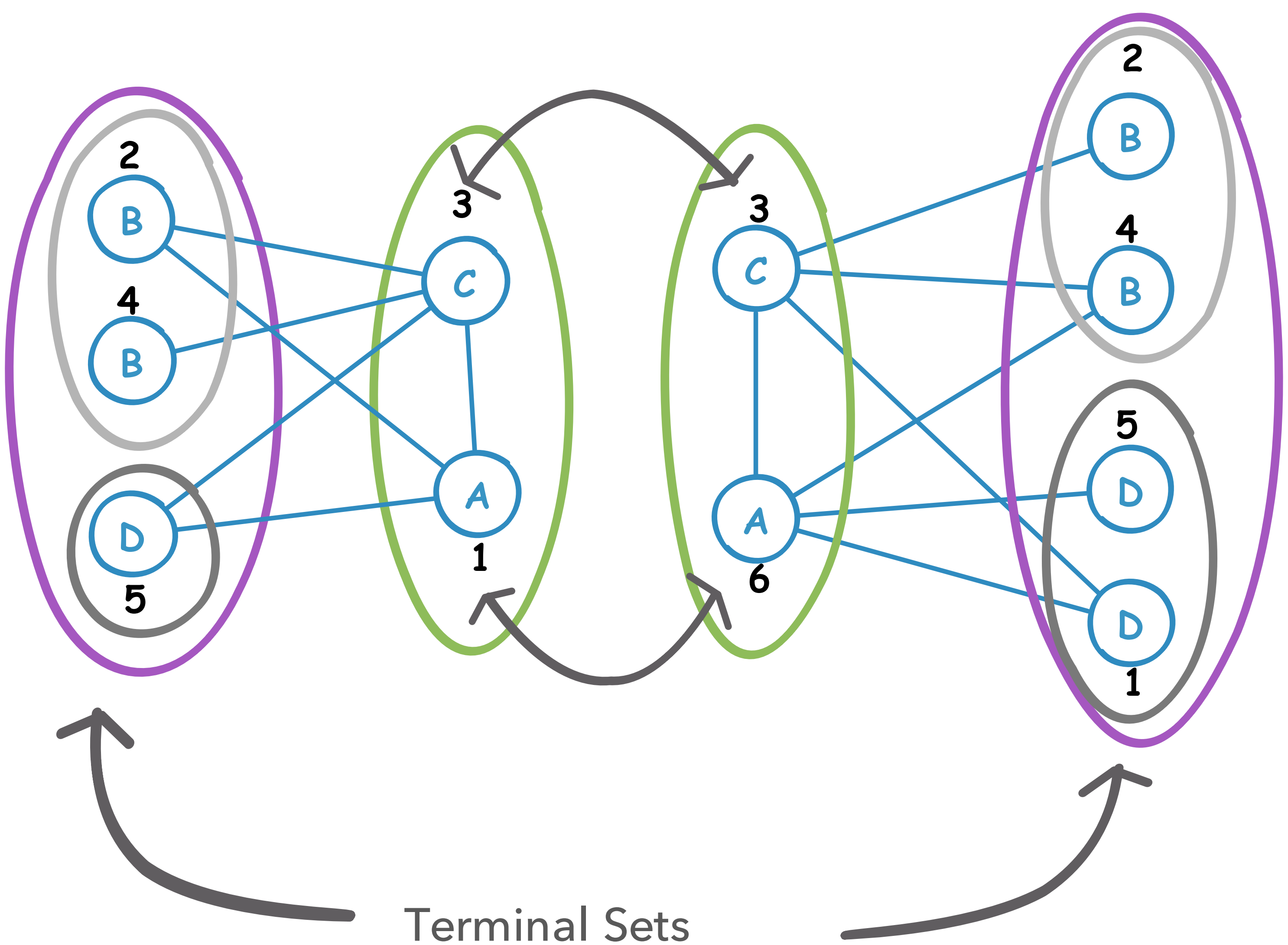


VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency?

Feasibility Rules

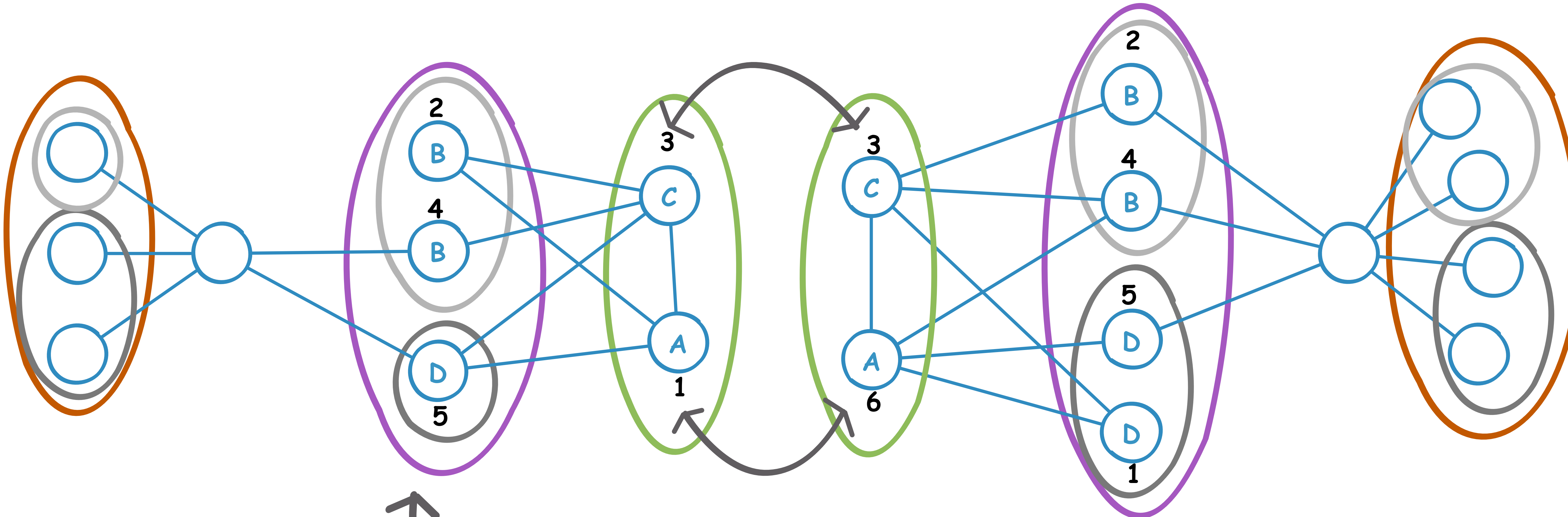
Look-ahead Rules



VF3: HOW DOES IT WORK? (FEASIBILITY)

How do we check for consistency? → Feasibility Rules

Look-ahead Rules



Terminal Sets


WWW.MIVIA.UNISA.IT