ARC CONSISTENCY WITH NEGATIVE VARIANT TABLES

Albert Haag
SAP SE, Germany
(partial-retirement)
email: albert.haag@t-online.de

Disclaimer:
All work on my papers “Arc Consistency with Negative Variant Table” and “Column Oriented Compilation of Variant Tables” was performed privately during the last two years after transition into partial retirement.

It and the accompanying implementation are neither endorsed by SAP nor do they reflect ongoing SAP development.

Notwithstanding:
The motivation for this work lies in my past at SAP and is based on insights and experiences with the SAP product configurators. The terminology follows that used in conjunction with the SAP Variant Configurator and originates from roots in the manufacture of „variants“.

See paper(s) for references

Negative Variant Table (NVTAB) $\mathcal{U}$

- List of exclusions/ disallowed combinations of product properties in tabular form

Negative Variant Table (NVTAB) $\mathcal{U}$

- List of exclusions/ disallowed combinations of product properties in tabular form
- Same example: t-shirt

Negative Variant Table (NVTAB) $\mathcal{U}$

- List of exclusions/ disallowed combinations of product properties in tabular form
- Same example: t-shirt

Color	Size	Print
Black	Small	STW (Save the Whales)
Red	Small	MIB (Men in Black)
Red	Medium	MIB
Red	Large	MIB
Red	Small	STW
White	Small	MIB
White	Medium	MIB
White	Large	MIB
White	Small	STW
Blue	Small	MIB
Blue	Medium	MIB
Blue	Large	MIB
Blue	Small	STW

Negative Variant Table (NVTAB) $\mathcal{U}$

- List of exclusions/ disallowed combinations of product properties in tabular form
- Same example: t-shirt
- Customer requirement

Color	Size	Print
Black	Small	STW (Save the Whales)
Red	Small	MIB (Men in Black)
Red	Medium	MIB
Red	Large	MIB
Red	Small	STW
White	Small	MIB
White	Medium	MIB
White	Large	MIB
White	Small	STW
Blue	Small	MIB
Blue	Medium	MIB
Blue	Large	MIB
Blue	Small	STW

Considerations about NVTABs

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Customer motivation for using NVTABs:

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Customer motivation for using NVTABs:

- Negative representation more compact
- Global domains may not be finite

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Customer motivation for using NVTABs:

- Negative representation more compact
- Global domains may not be finite
- <u>Independence of changes to global domains</u>

Considerations about NVTABs

Theoretically identical to a positive table representing complement to a (finite) global domain

Customer motivation for using NVTABs:

- Negative representation more compact
- Global domains may not be finite
- <u>Independence of changes to global domains</u>
  - Value domains may change often (typically daily)
  - Hope that NVTABs are less sensitive to these changes

Use of NVTABs in Configuration

Use of NVTABs in Configuration

Arc-consistency (constraint propagation): Eliminate all values that do not have support from each column domain

Use of NVTABs in Configuration

Arc-consistency (constraint propagation): Eliminate all values that do not have support from each column domain

➔ Here, I deal with arc-consistency only

Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

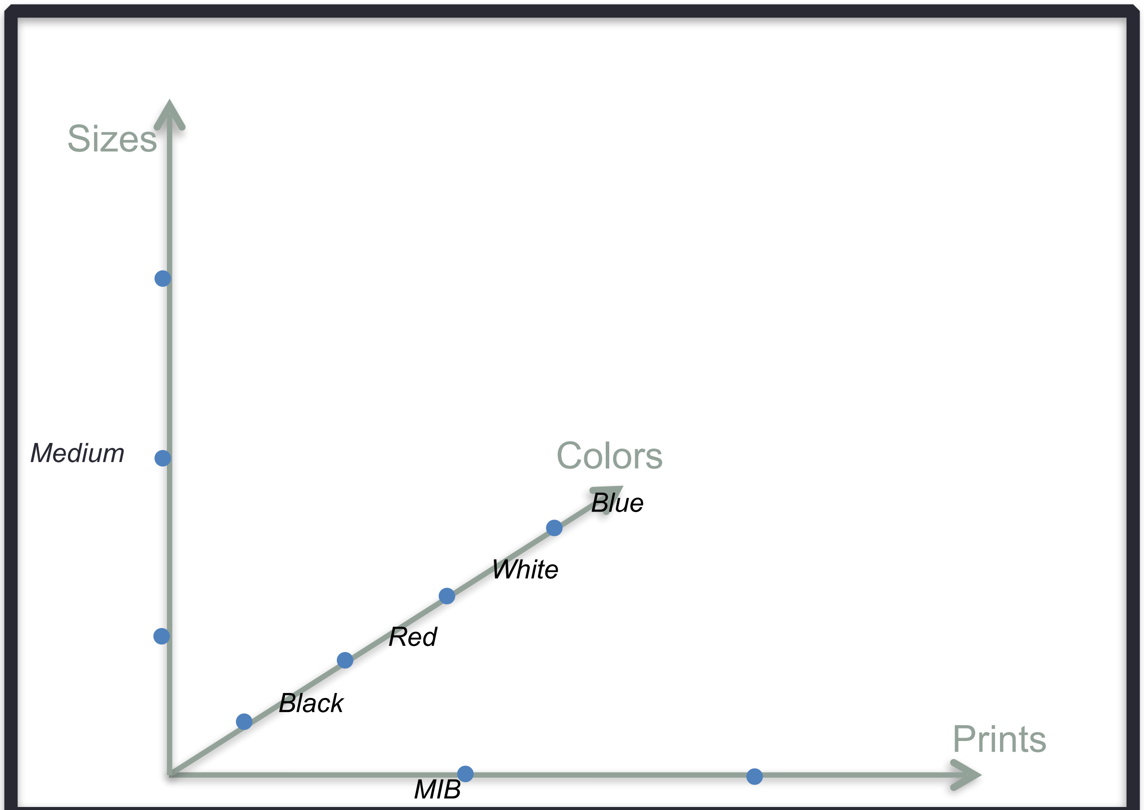
- Given the k product properties of a VTAB:

Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

- Given the *k* product properties of a VTAB:
  - The “solution space” is a *k*-dimensional cuboid (Cartesian product, c-tuple, …)

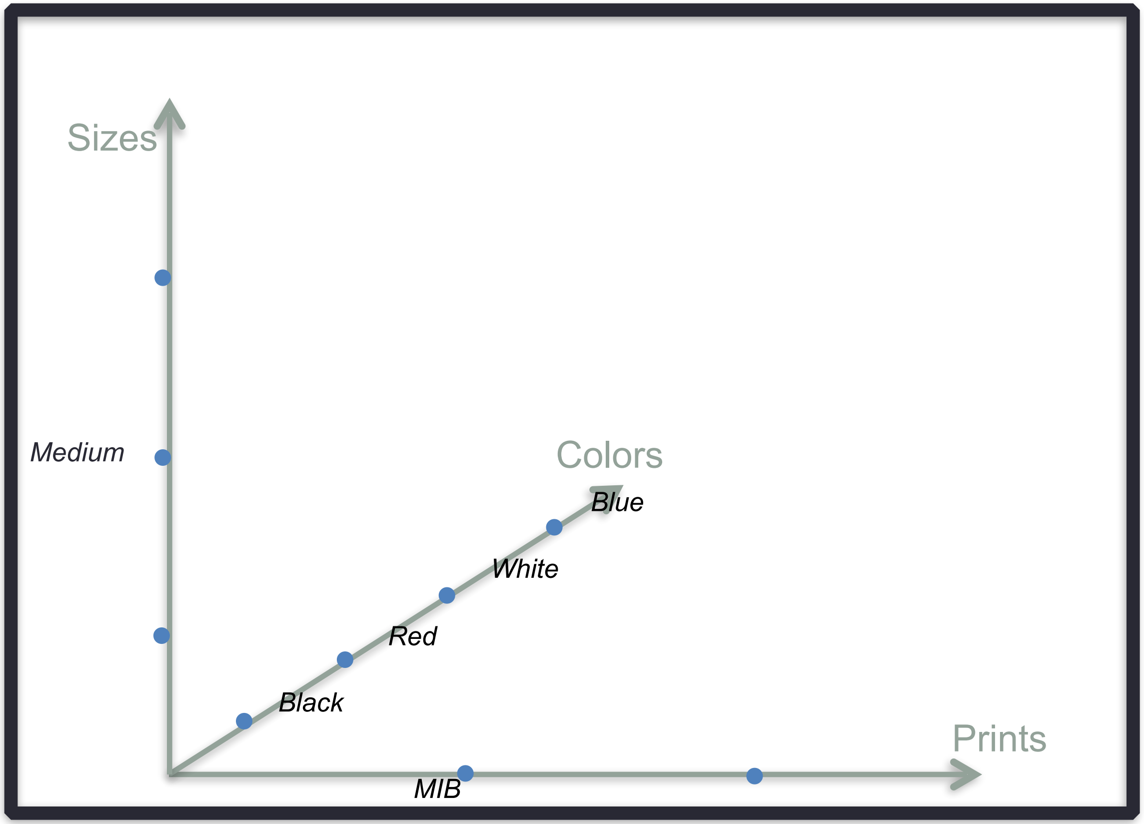
Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

- Given the *k* product properties of a VTAB:
  - The “solution space” is a *k*-dimensional cuboid (Cartesian product, c-tuple, …)



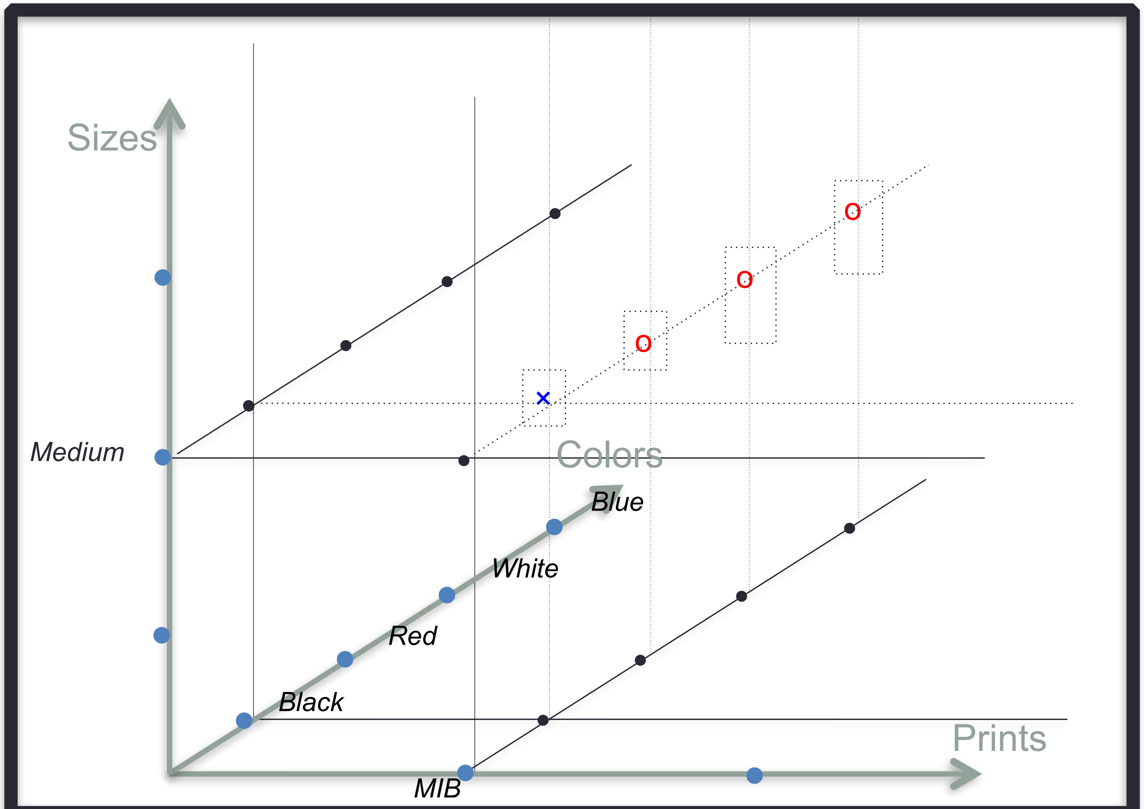
Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

- Given the *k* product properties of a VTAB:
  - The “solution space” is a *k*-dimensional cuboid (Cartesian product, c-tuple, …)
  - A VTAB/NVTAB is a (finite) subset of the “solution space”



Set-Theoretic Look at VTABs/NVTABs and Arc-consistency

- Given the *k* product properties of a VTAB:
  - The “solution space” is a *k*-dimensional cuboid (Cartesian product, c-tuple, …)
  - A VTAB/NVTAB is a (finite) subset of the “solution space”



Legend
x – row in table
o – row not in table,

	Black	Red	White	Blue
Large/MIB	*x*	*o*	*o*	*o*
Large/STW	*x*	*x*	*x*	*x*
Medium/MIB	*x*	*o*	*o*	*o*
Medium/STW	*x*	*x*	*x*	*x*
Small/MIB	*x*	*o*	*o*	*o*
Small/STW	*o*	*o*	*o*	*o*

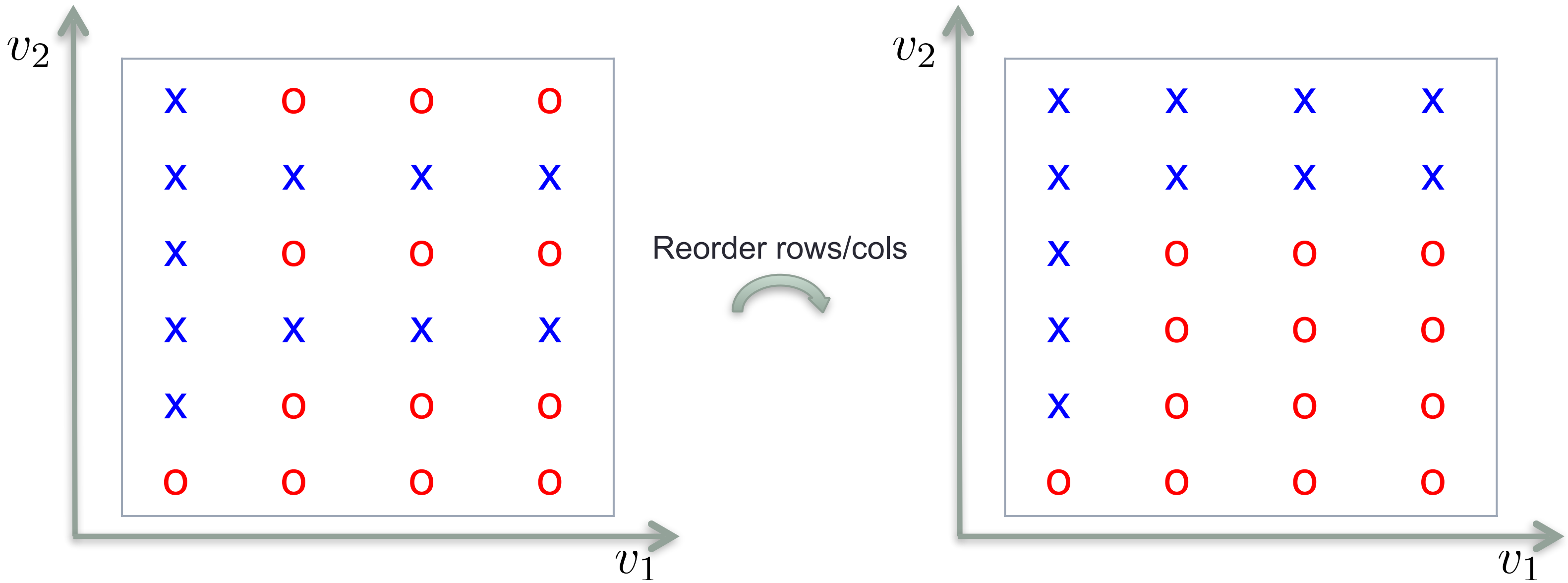
Two-dimensional depiction of valid t-shirts

Solution space: (2x3) x 4 = 24 rows

Legend

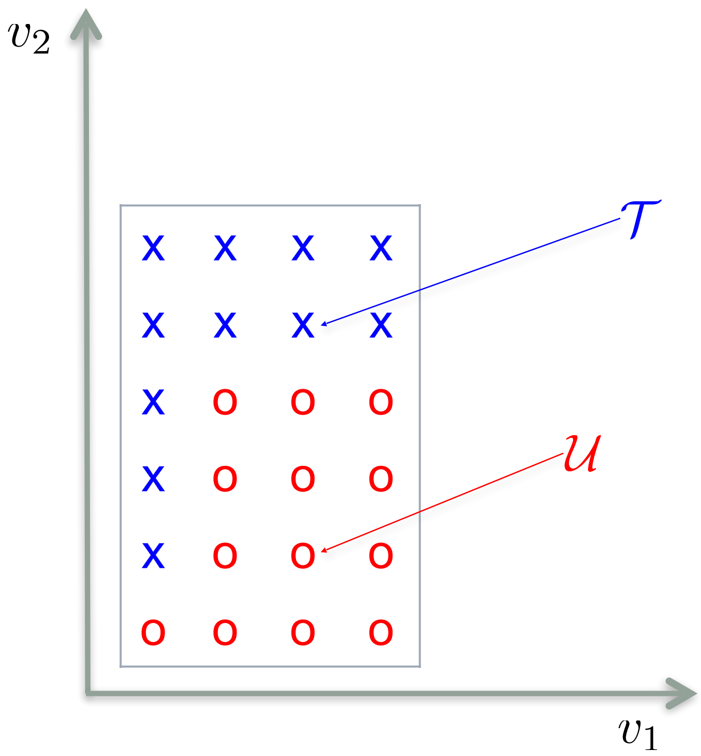
x – row in table

o – row not in table

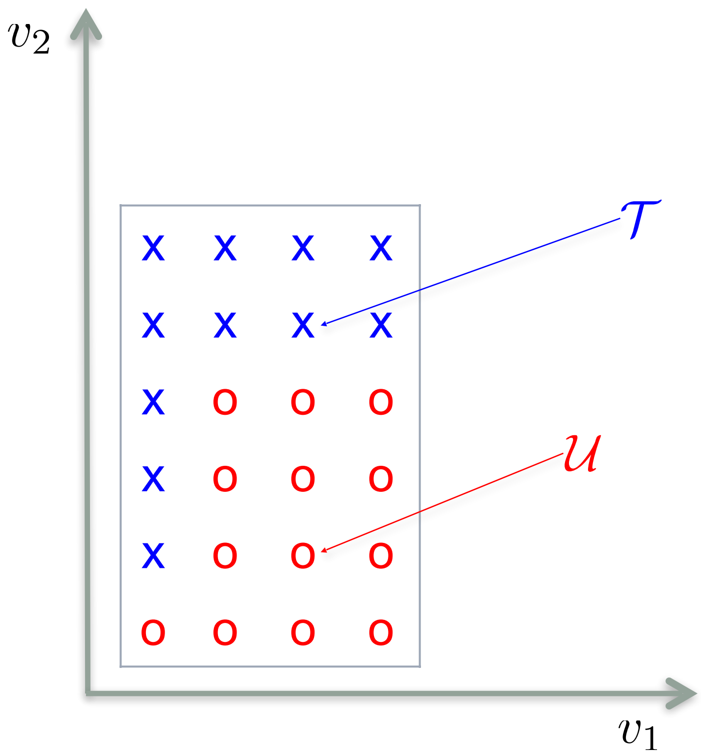


Abstract depiction of t-shirt VTAB/NVTAB and reordering

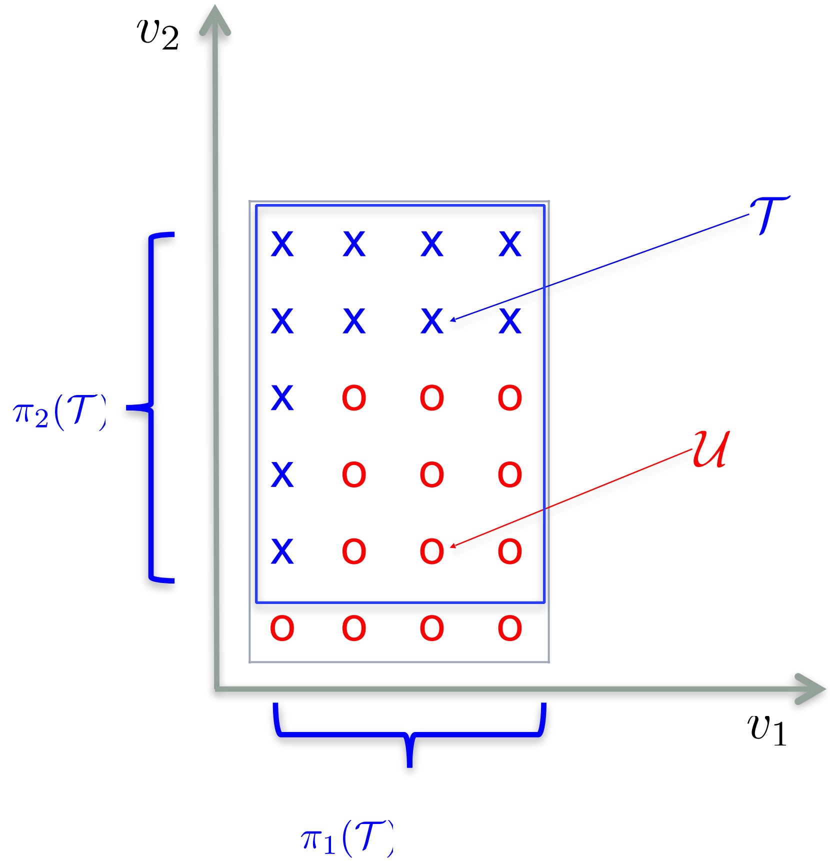
For VTAB everything outside depicted area is “x” (allowed combination)
For VTAB everything outside depicted area is “o” (dis-allowed combination)



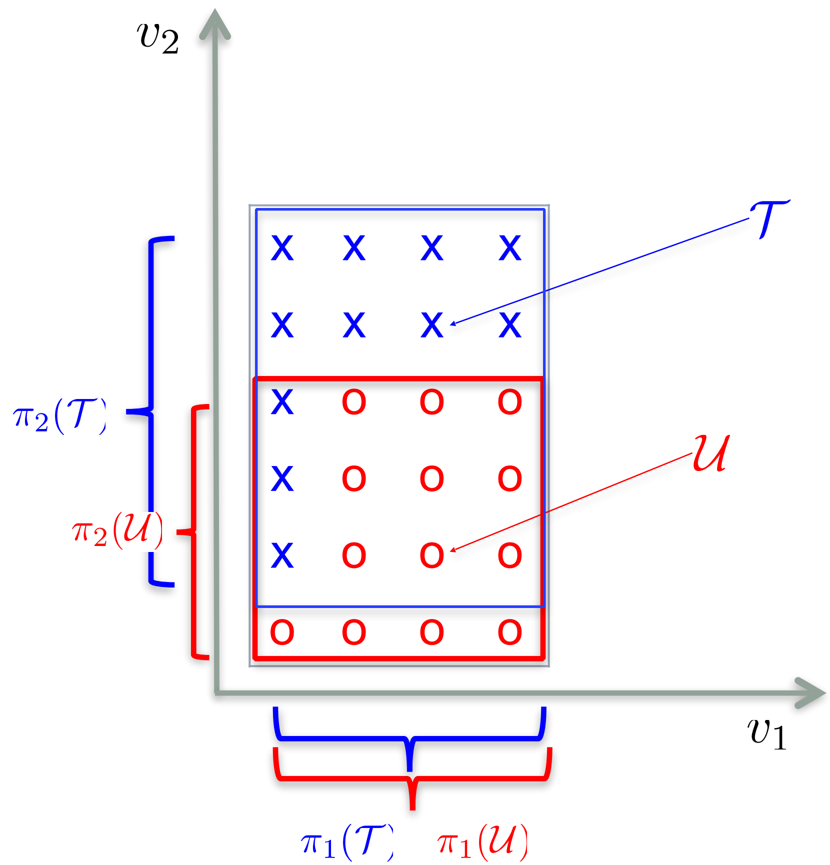
v_2
$\mathcal{T}$
x x x x
x x x x
x o o o
x o o o
x o o o
o o o o
$\mathcal{U}$
v_1



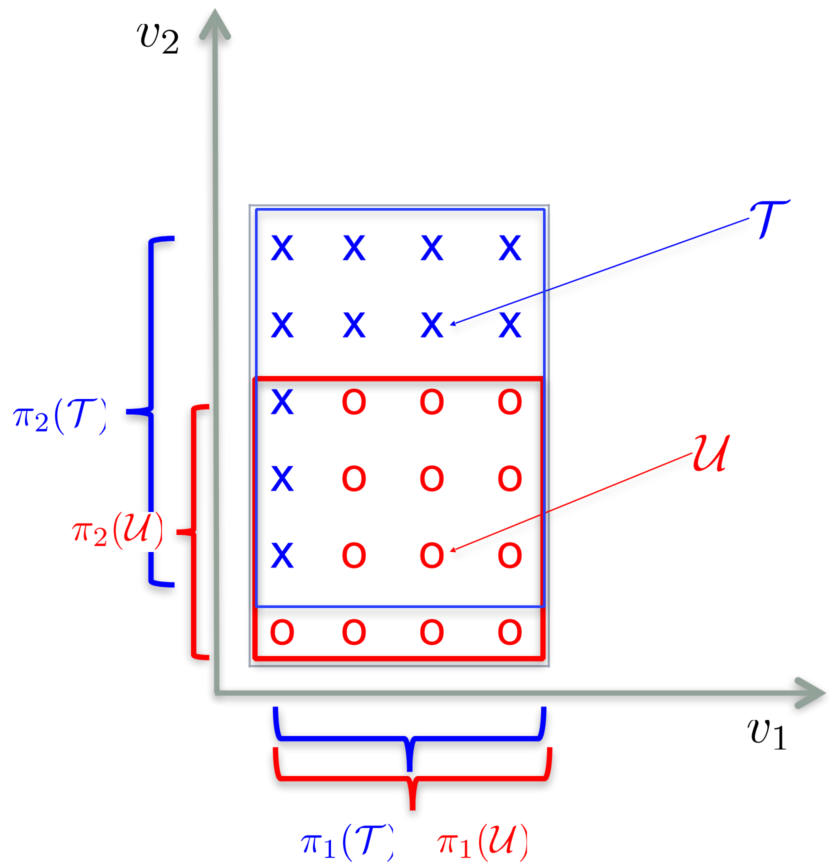
Projections onto the *k*-axes and supported values



Projections onto the *k*-axes and supported values



Projections onto the *k*-axes and supported values



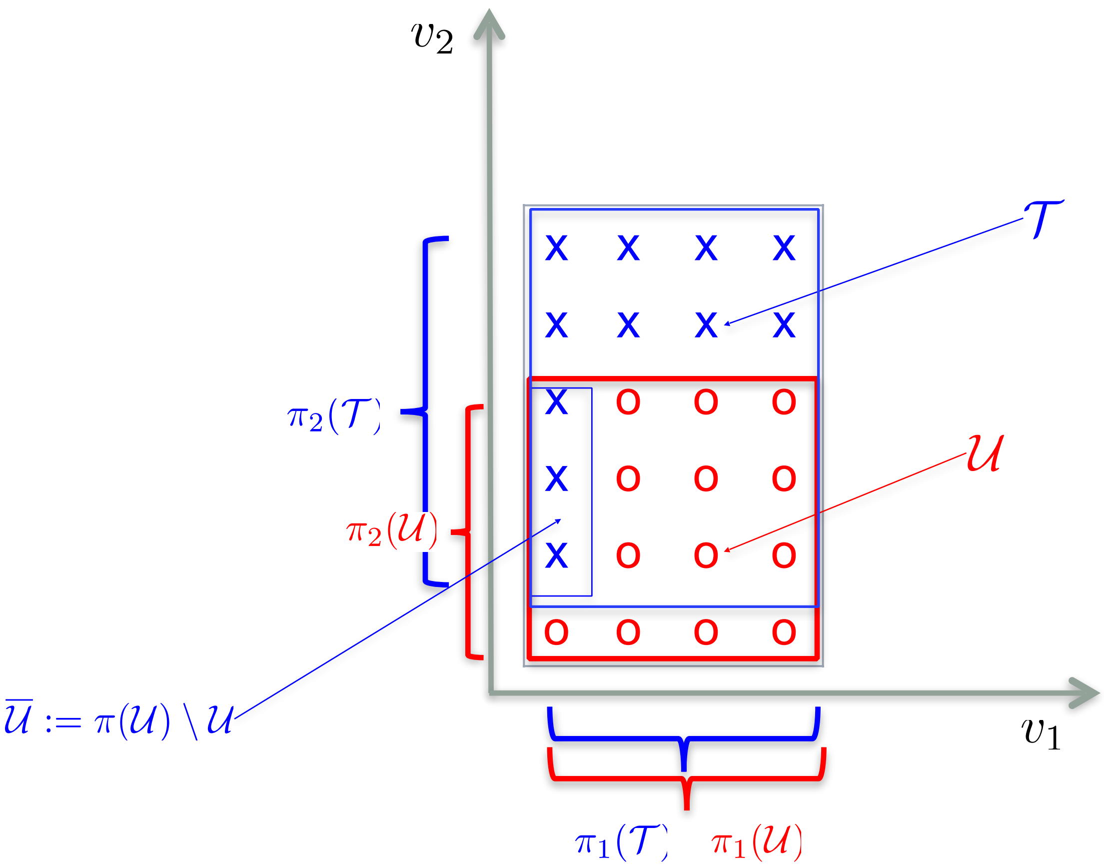
Projections onto the *k*-axes and supported values

$\pi_j(\mathcal{T})$ – projection of valid rows onto *j*-th axis

$\pi_j(\mathcal{U})$ – projection of invalid rows onto *j*-th axis

A value in $\pi_j(\mathcal{T})$ has support (is arc-consistent)

$\bar{\mathcal{U}} = \pi(\mathcal{U}) \backslash \mathcal{U}$ is positive VTAB



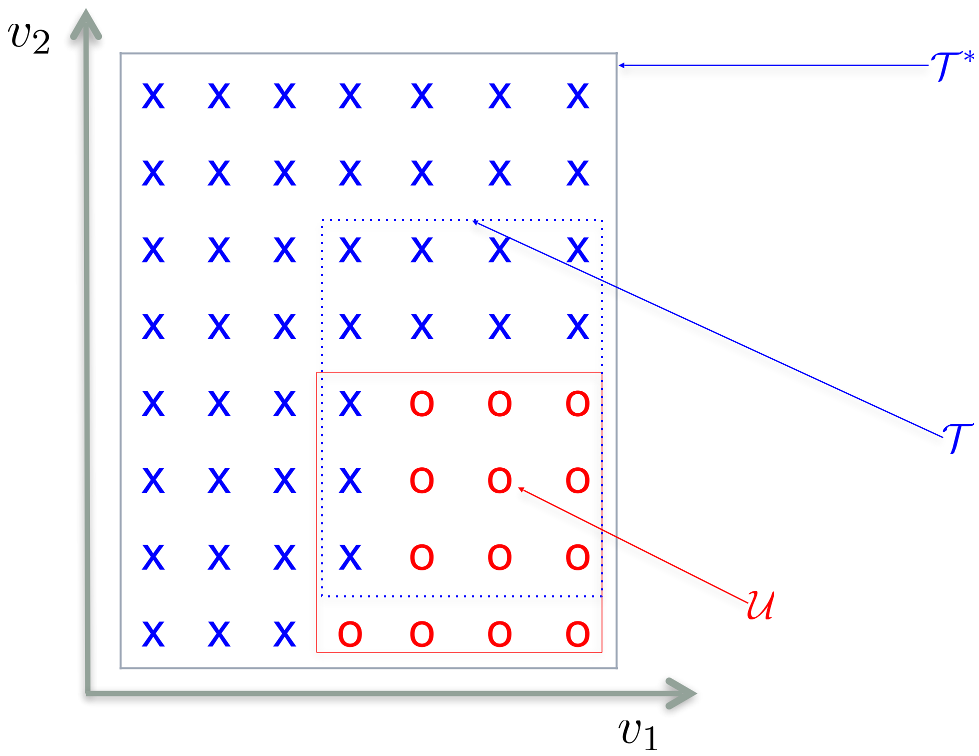
Projections onto the *k*-axes and supported values

$\pi_j(\mathcal{T})$ – projection of valid rows onto *j*-th axis

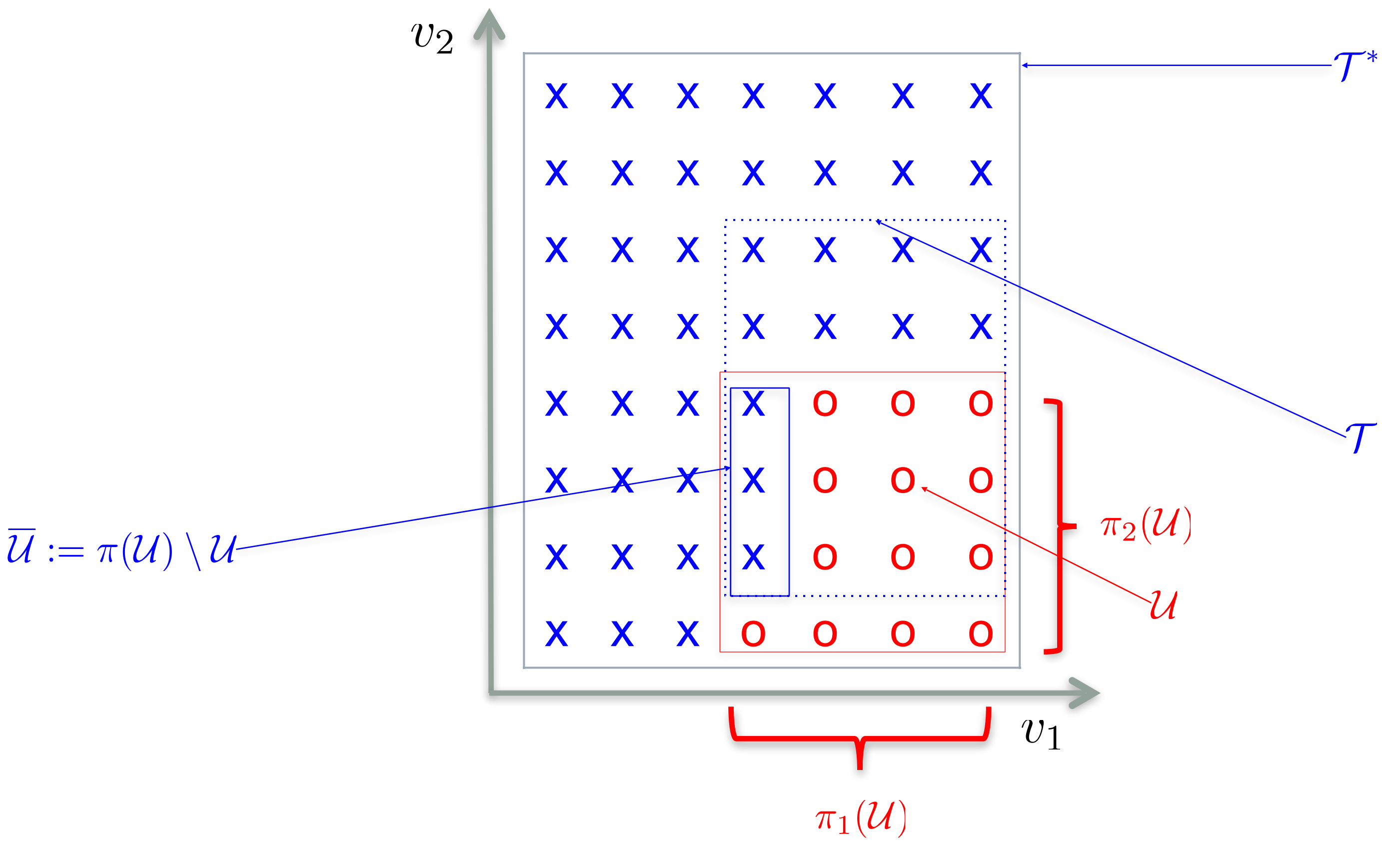
$\pi_j(\mathcal{U})$ – projection of invalid rows onto *j*-th axis

A value in $\pi_j(\mathcal{T})$ has support (is arc-consistent)

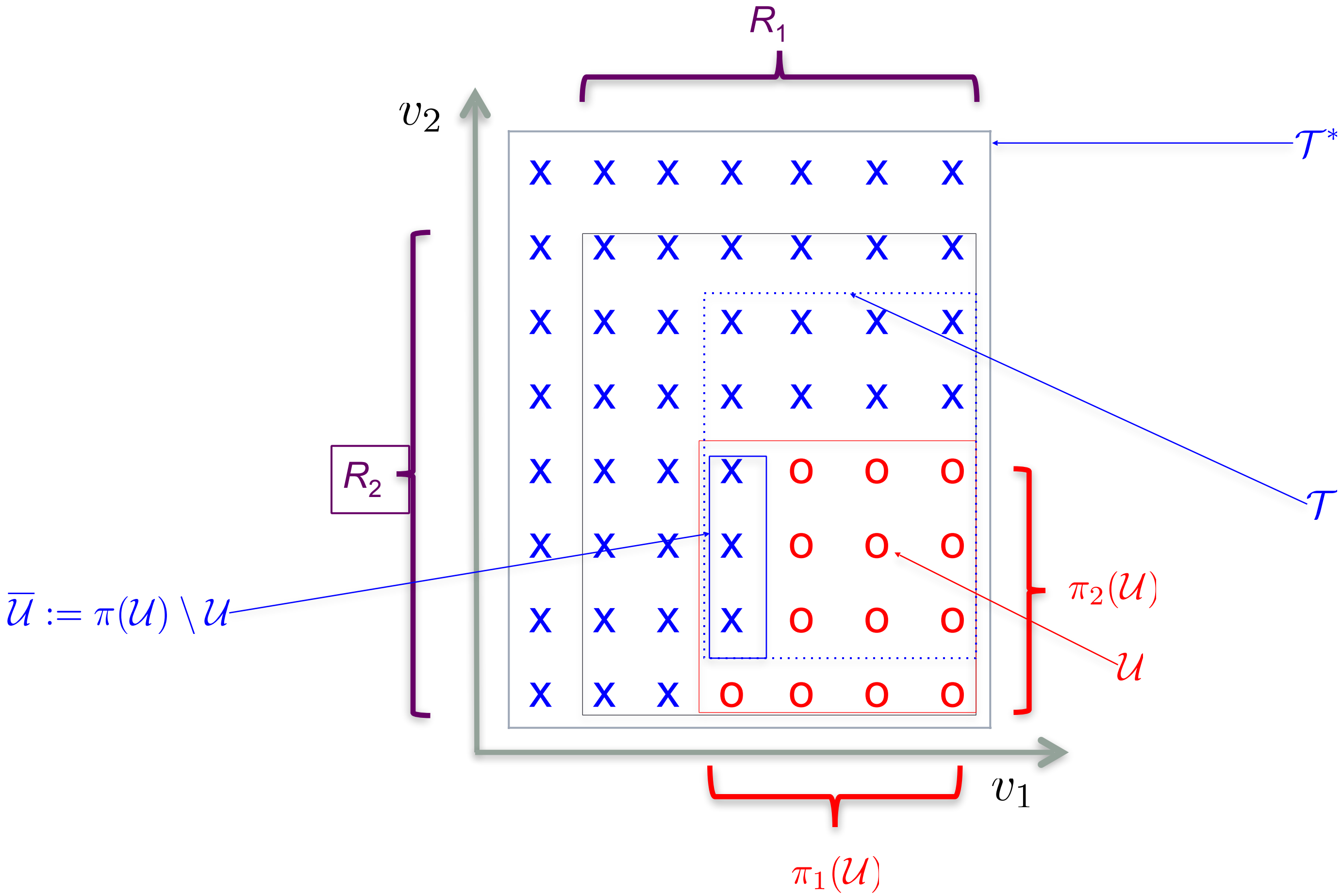
$\bar{\mathcal{U}} = \pi(\mathcal{U}) \setminus \mathcal{U}$ is positive VTAB



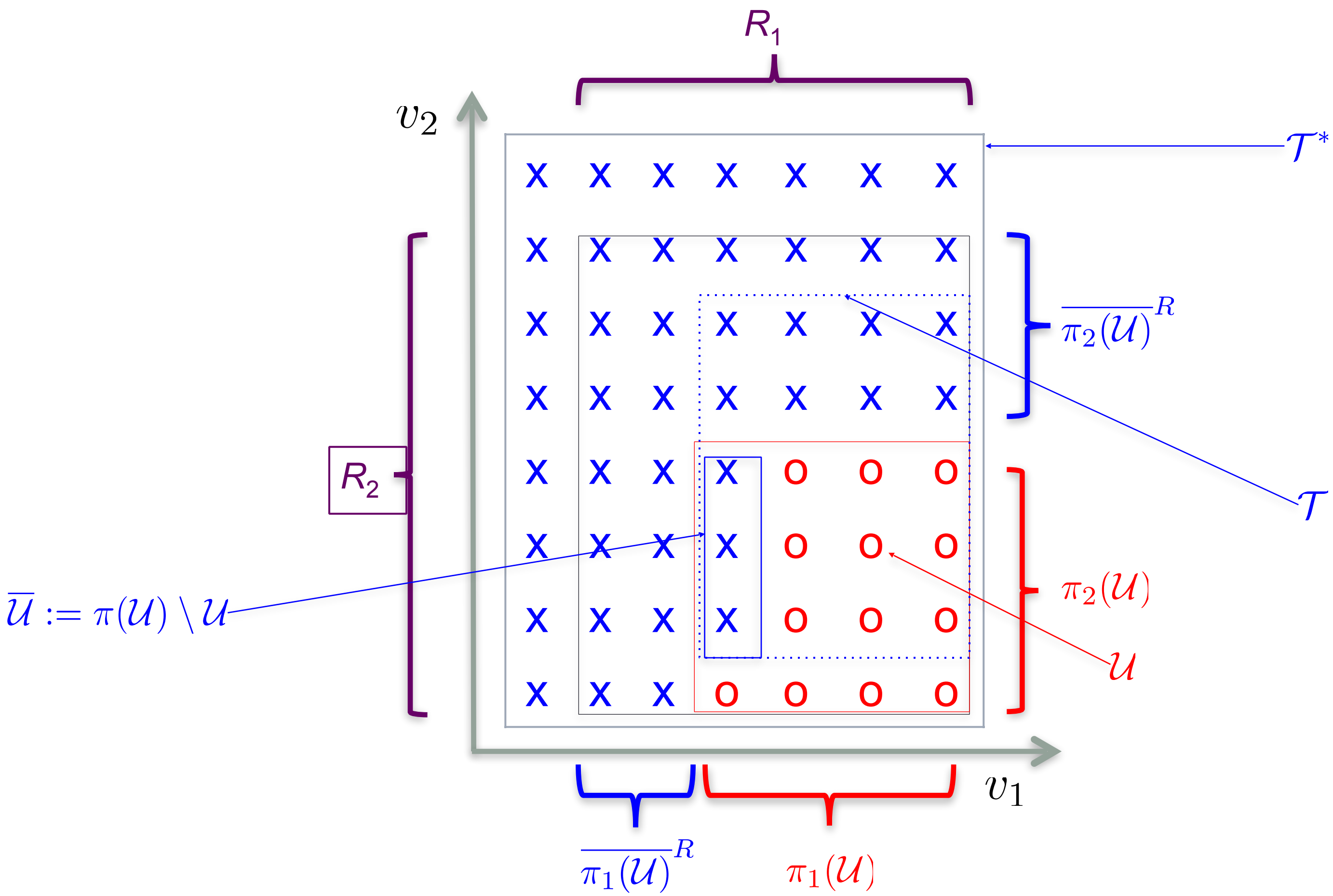
$\mathcal{T}^*$: Adding more values to v_1 and v_2 while leaving NVTAB $\mathcal{U}$ unchanged



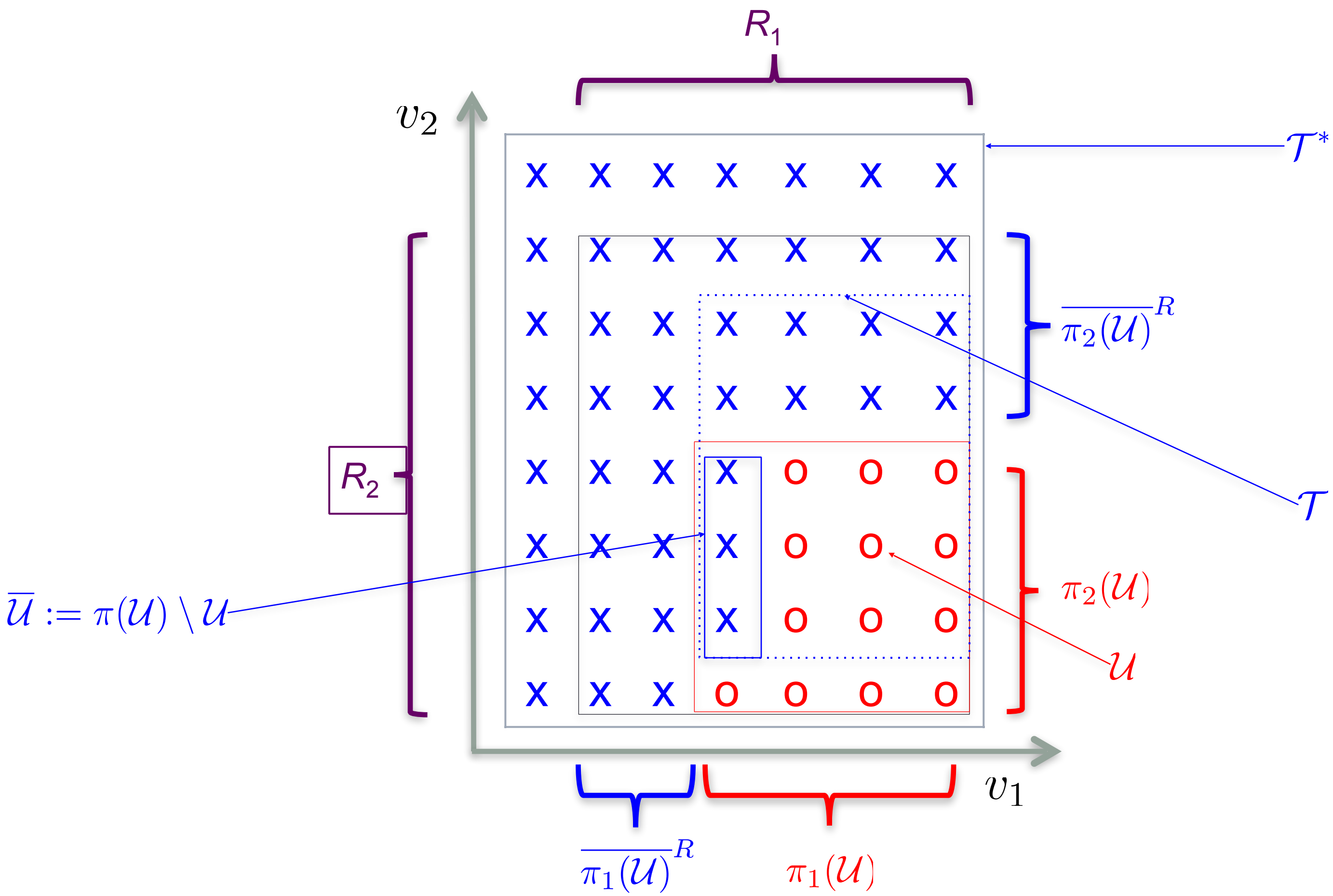
$\mathcal{T}^*$: Adding more values to v_1 and v_2 while leaving NVTAB $\mathcal{U}$ unchanged



$\mathcal{T}^*$: Adding more values to v_1 and v_2 while leaving NVTAB $\mathcal{U}$ unchanged



$\mathcal{T}^*$: Adding more values to v_1 and v_2 while leaving NVTAB $\mathcal{U}$ unchanged



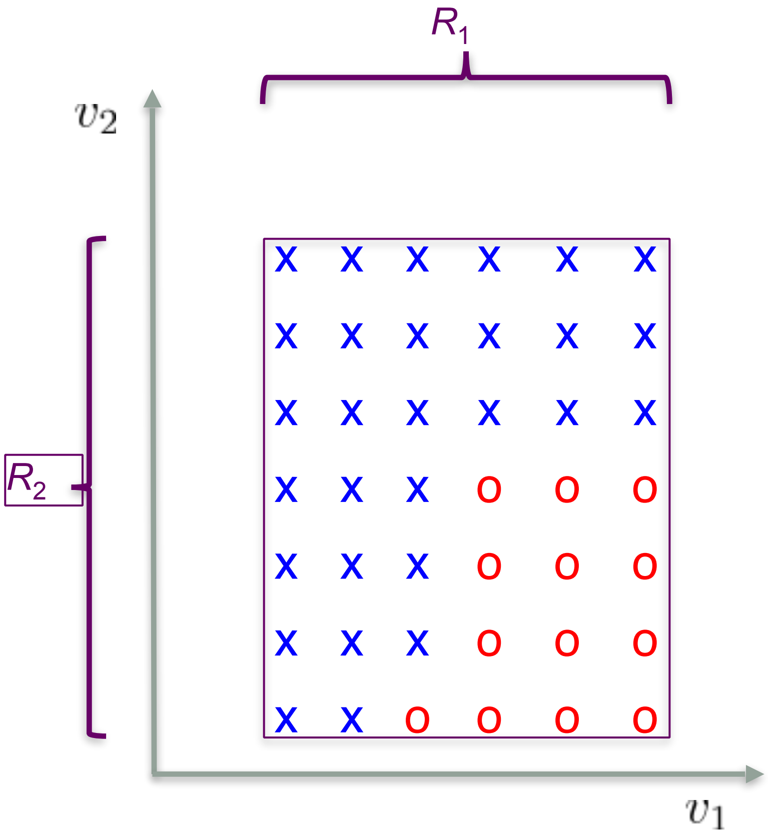
$\mathcal{T}^*$: Adding more values to v_1 and v_2 while leaving NVTAB $\mathcal{U}$ unchanged

$\pi_j(\mathcal{U})$ – projection of NVTAB $\mathcal{U}$ onto j-th axis (determined at maintenance time)

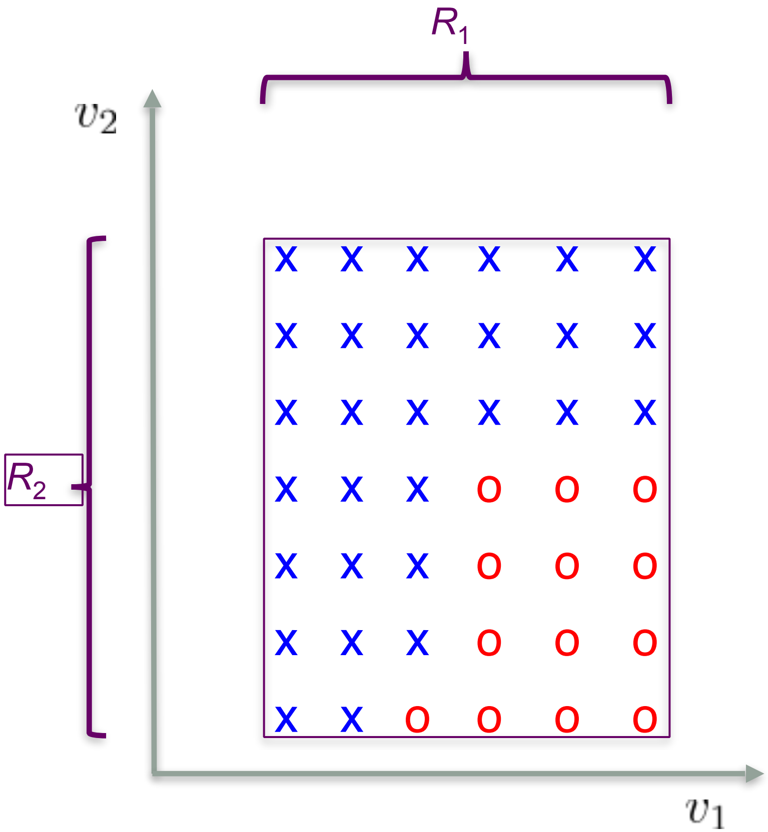
R_j – restrictions known at run-time (time of evaluation)

$\overline{\pi_j(\mathcal{U})}^R$ – complement of $\pi_j(\mathcal{U})$ to R (determined at run-time)

Obs#1 – Decomposition of Solution Set

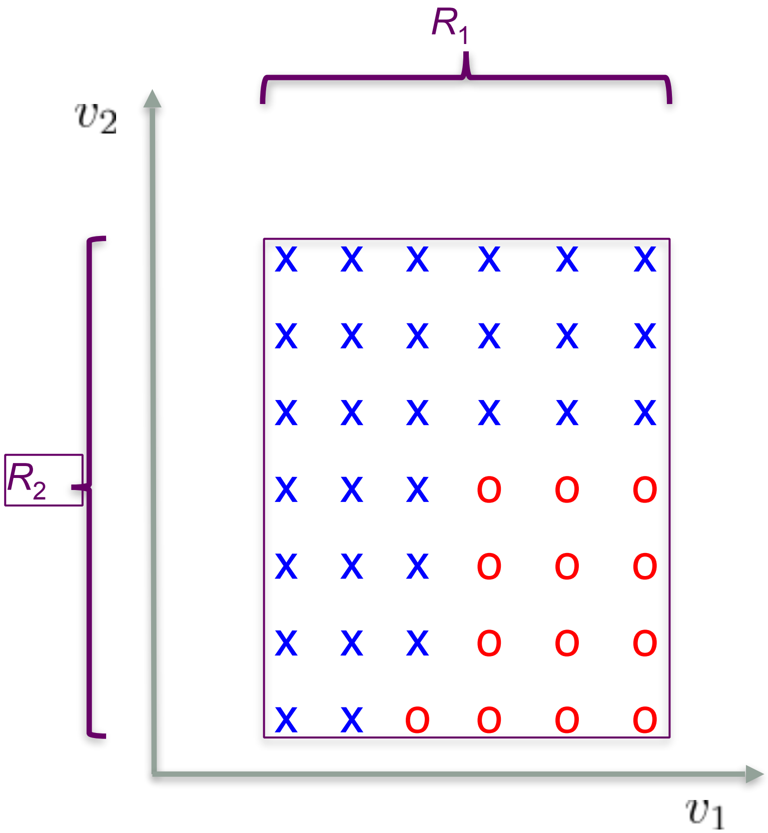


Obs#1 – Decomposition of Solution Set



Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

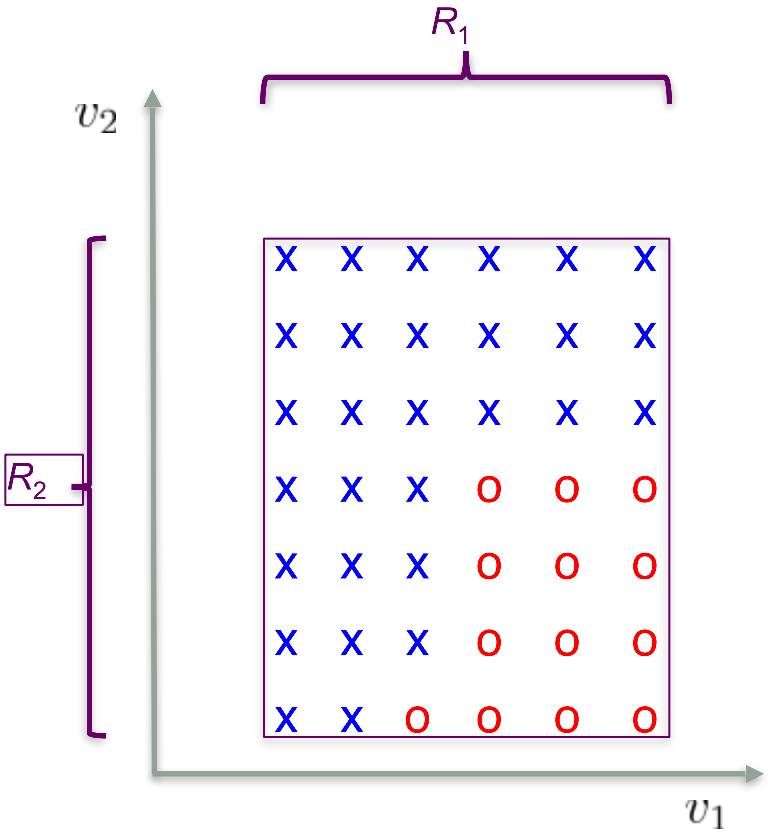
Obs#1 – Decomposition of Solution Set



Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

Obs#1 – Decomposition of Solution Set

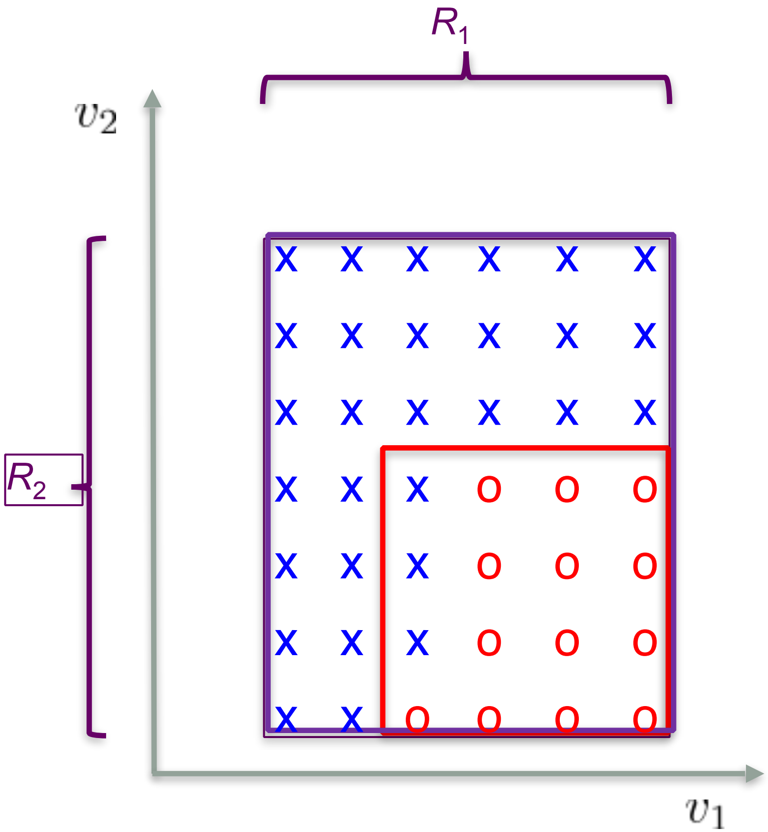


Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

- Difference between two cuboids

Obs#1 – Decomposition of Solution Set

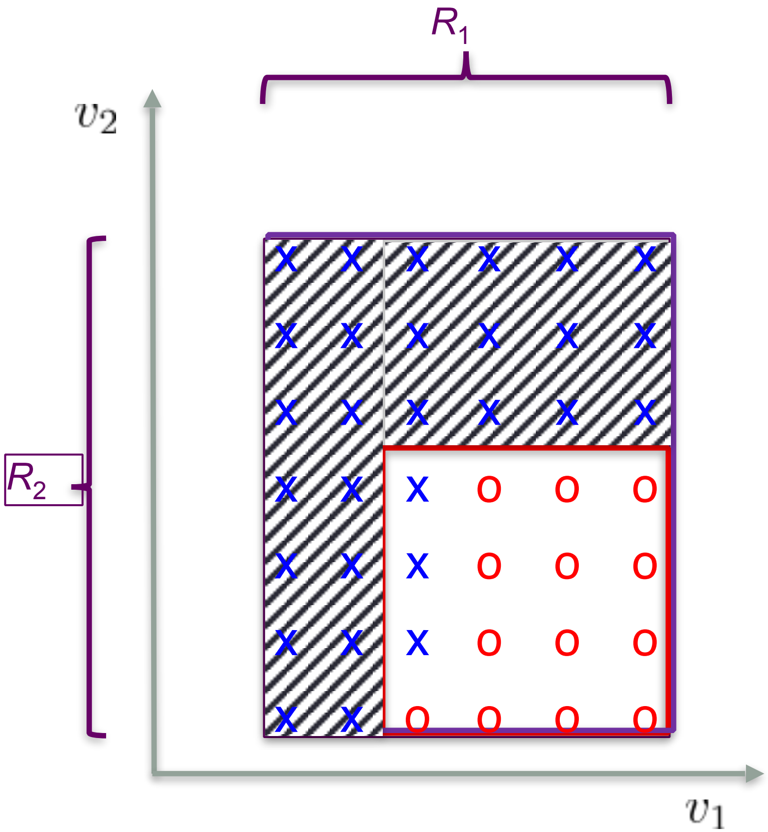


Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

- Difference between two cuboids

Obs#1 – Decomposition of Solution Set

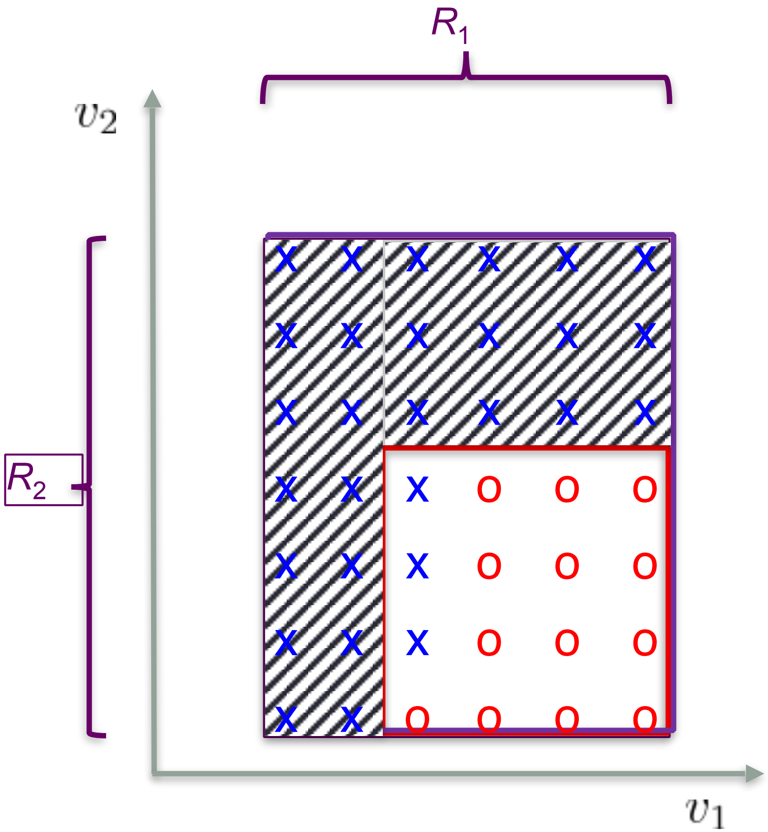


Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

- Difference between two cuboids

Obs#1 – Decomposition of Solution Set

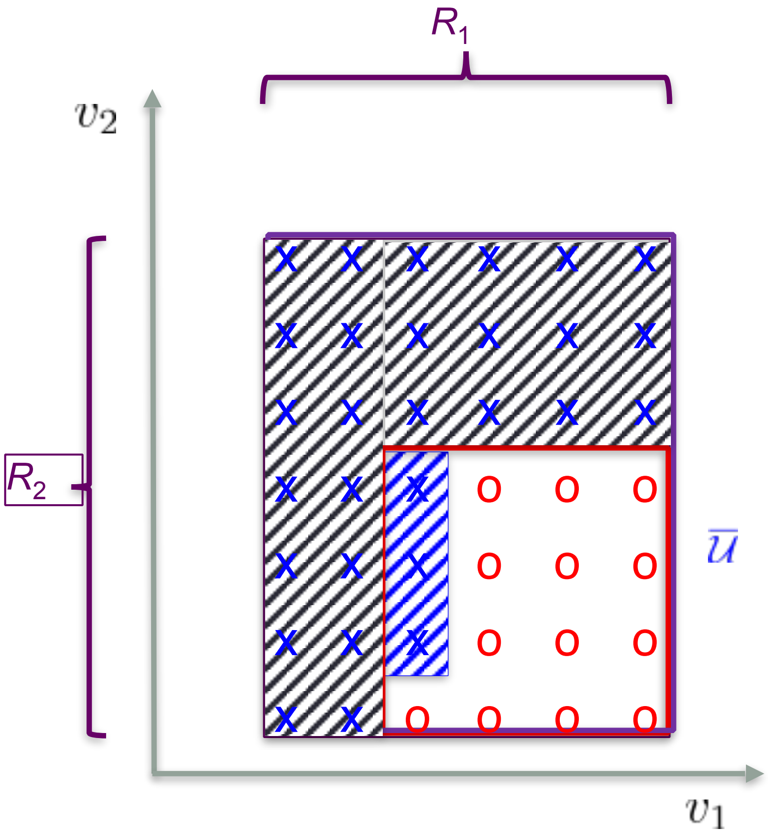


Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

- Difference between two cuboids
- Adding $\overline{\mathcal{U}}$

Obs#1 – Decomposition of Solution Set

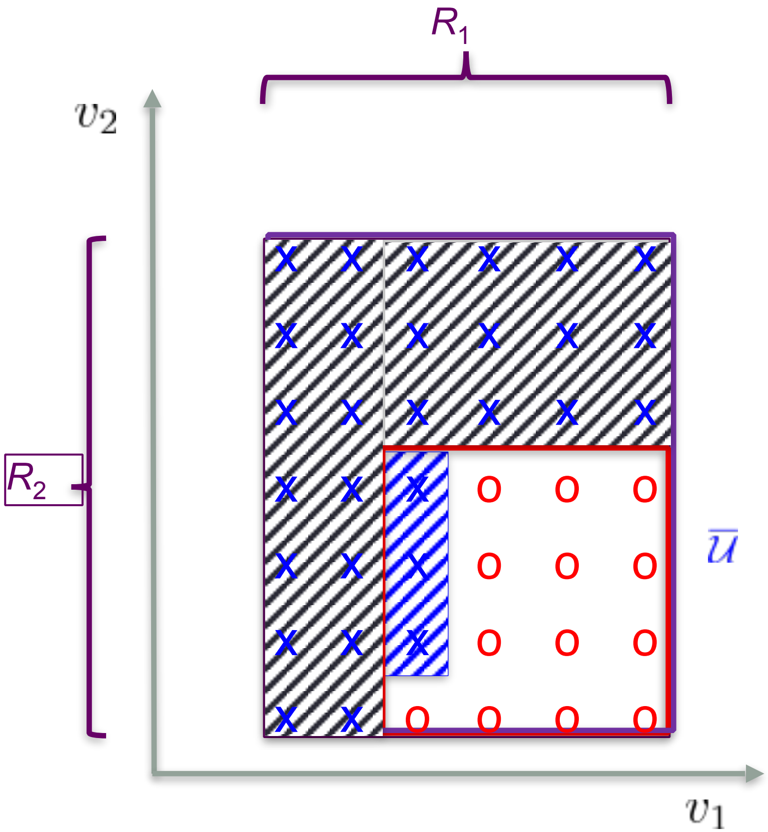


Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

- Difference between two cuboids
- Adding $\overline{\mathcal{U}}$

Obs#1 – Decomposition of Solution Set



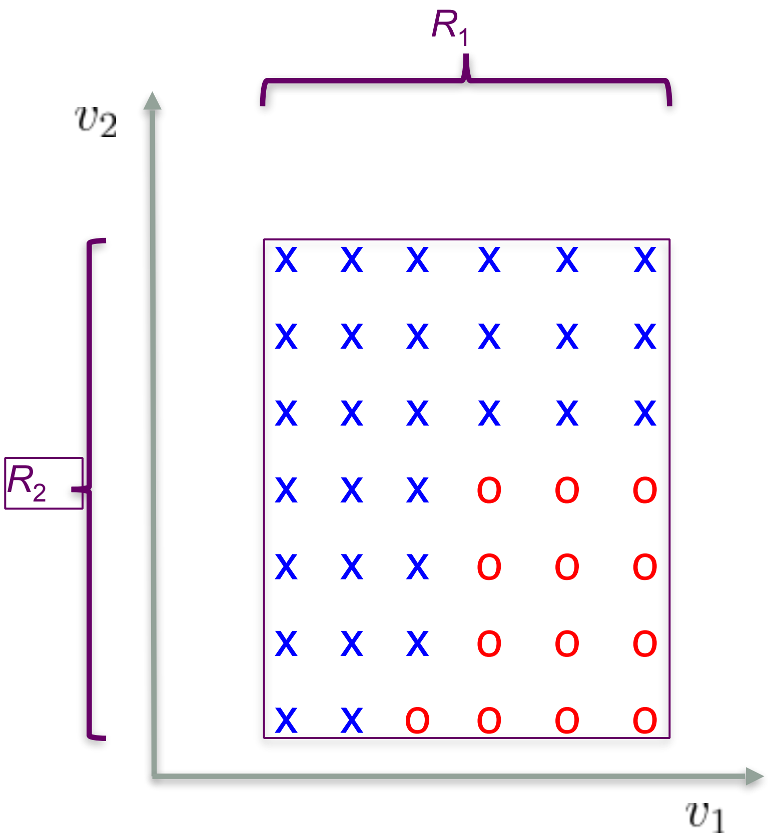
Solution set is defined as $\mathcal{S}^{\mathcal{U},R} := R \setminus \mathcal{U}$

Solution set has two parts:

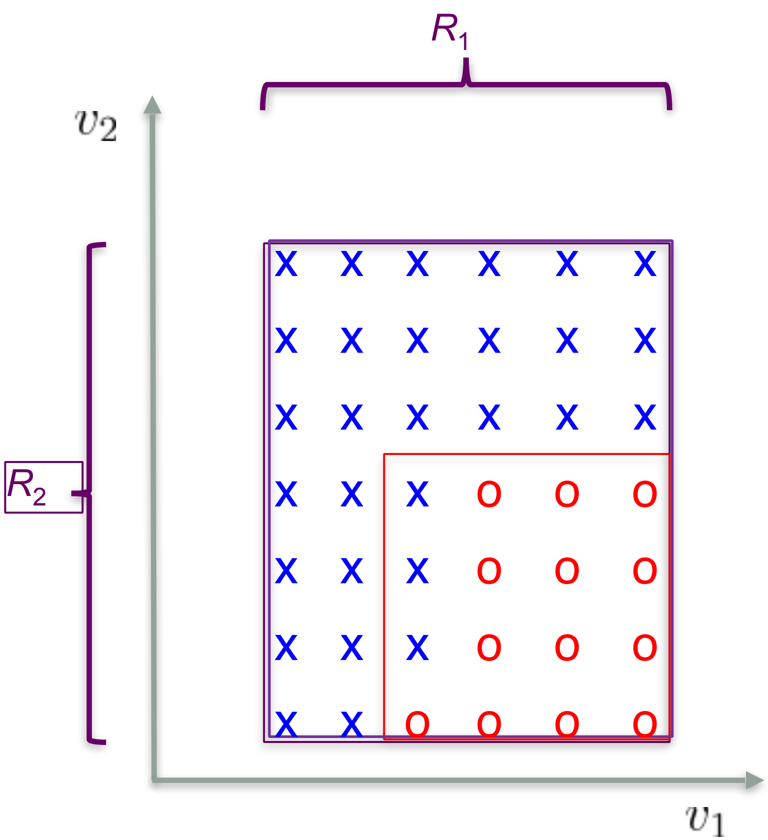
- Difference between two cuboids
- Adding $\overline{\mathcal{U}}$

$$\mathcal{S}^{\mathcal{U},R} := (R \setminus \pi(\mathcal{U})) \cup (\overline{\mathcal{U}} \cap R)$$

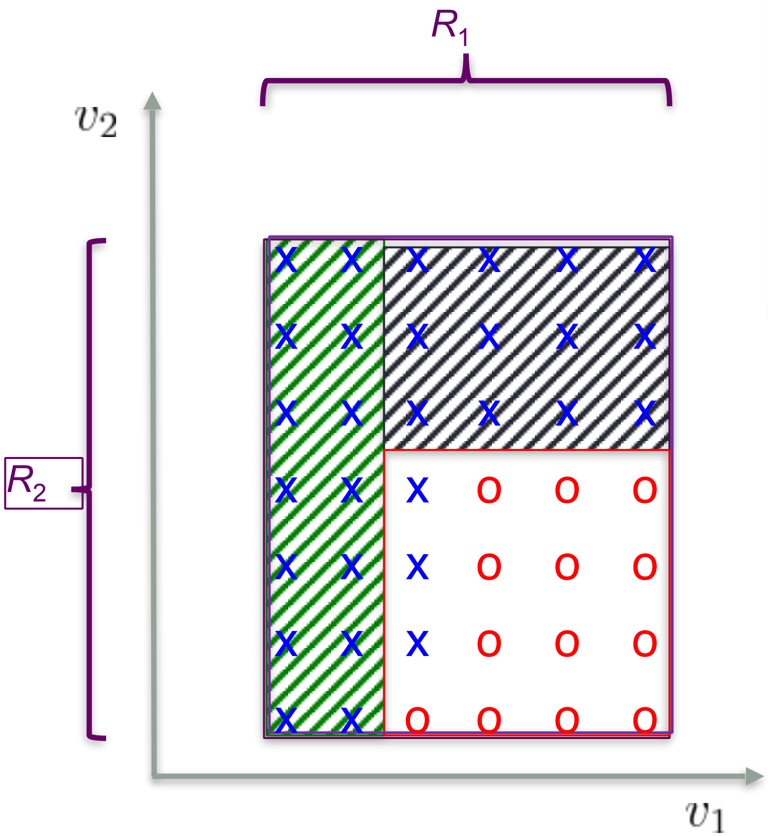
Disjoint Decomposition of $R \setminus \pi(\mathcal{U})$



Disjoint Decomposition of $R \setminus \pi(\mathcal{U})$



Disjoint Decomposition of $R \setminus \pi(\mathcal{U})$



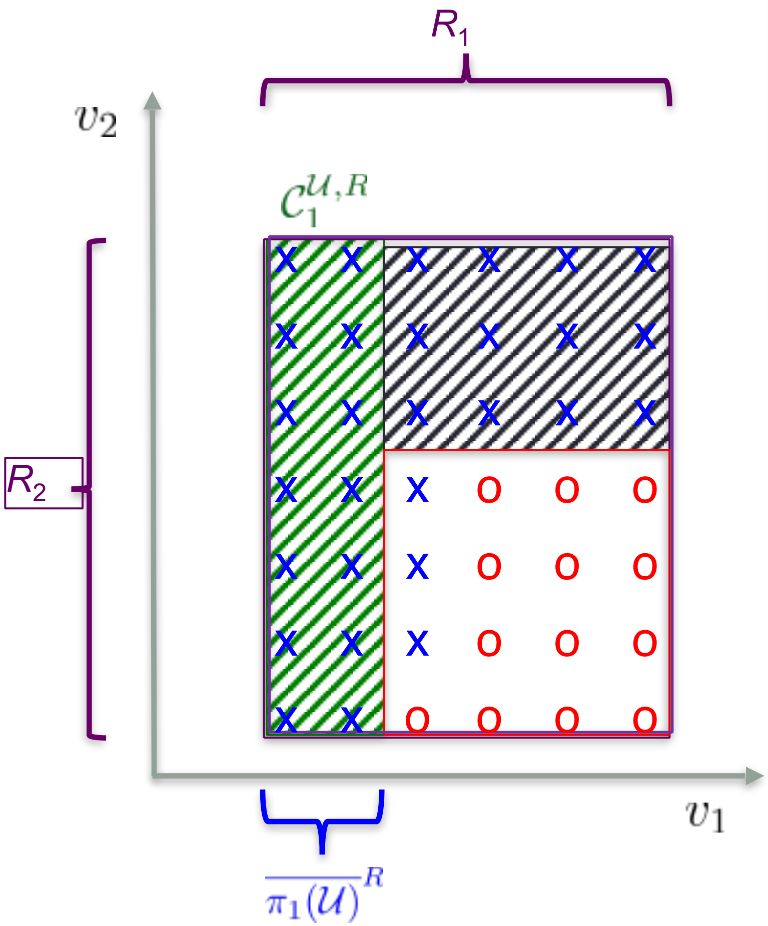
$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi_1(\mathcal{U})}^R \times R_2 \times R_3 \times \ldots \times R_k$$

$$\mathcal{C}_2^{\mathcal{U},R} = \pi_1(\mathcal{U}) \times \overline{\pi_2(\mathcal{U})}^R \times R_3 \times \ldots \times R_k$$

$$\ldots$$

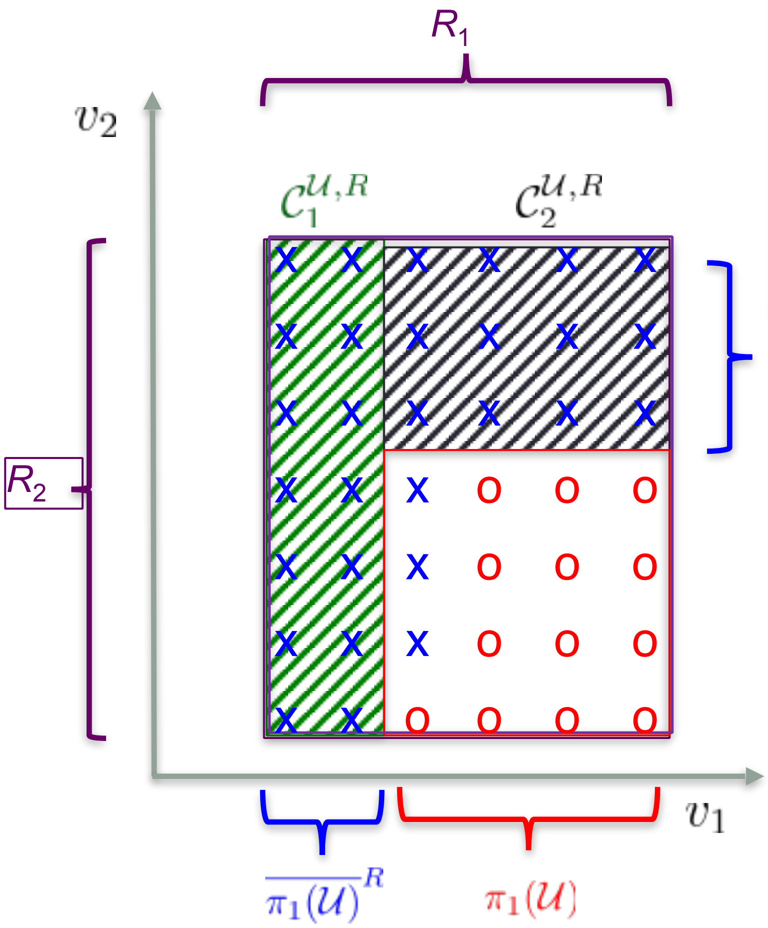
$$\mathcal{C}_k^{\mathcal{U},R} = \pi_1(\mathcal{U}) \times \pi_2(\mathcal{U}) \times \pi_3(\mathcal{U}) \times \ldots \times \overline{\pi_k(\mathcal{U})}^R$$

Disjoint Decomposition of $R \setminus \pi(\mathcal{U})$



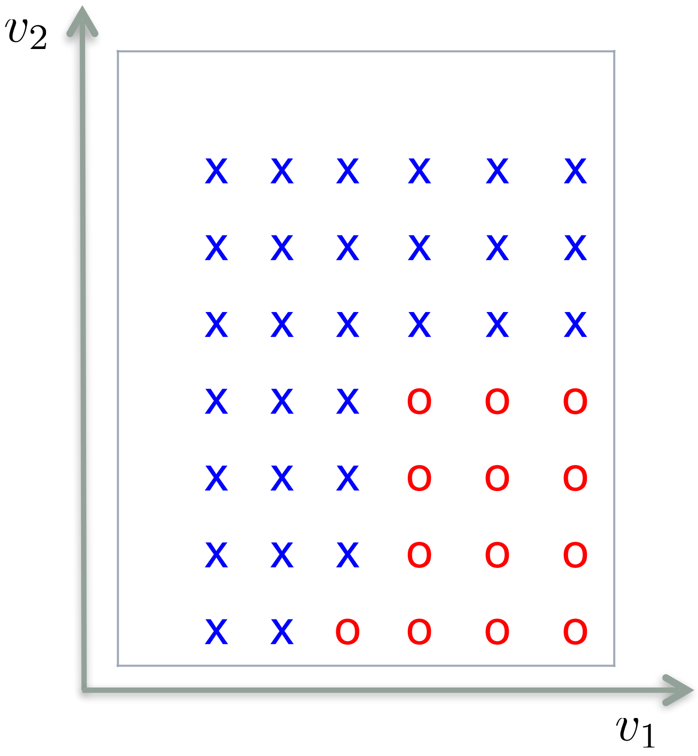
$$
\begin{aligned}
\mathcal{C}_1^{\mathcal{U},R} &= \overline{\pi_1(\mathcal{U})}^R \times R_2 \times R_3 \times \ldots \times R_k \\
\mathcal{C}_2^{\mathcal{U},R} &= \pi_1(\mathcal{U}) \times \overline{\pi_2(\mathcal{U})}^R \times R_3 \times \ldots \times R_k \\
&\ldots \\
\mathcal{C}_k^{\mathcal{U},R} &= \pi_1(\mathcal{U}) \times \pi_2(\mathcal{U}) \times \pi_3(\mathcal{U}) \times \ldots \times \overline{\pi_k(\mathcal{U})}^R
\end{aligned}
$$

Disjoint Decomposition of $R \setminus \pi(\mathcal{U})$

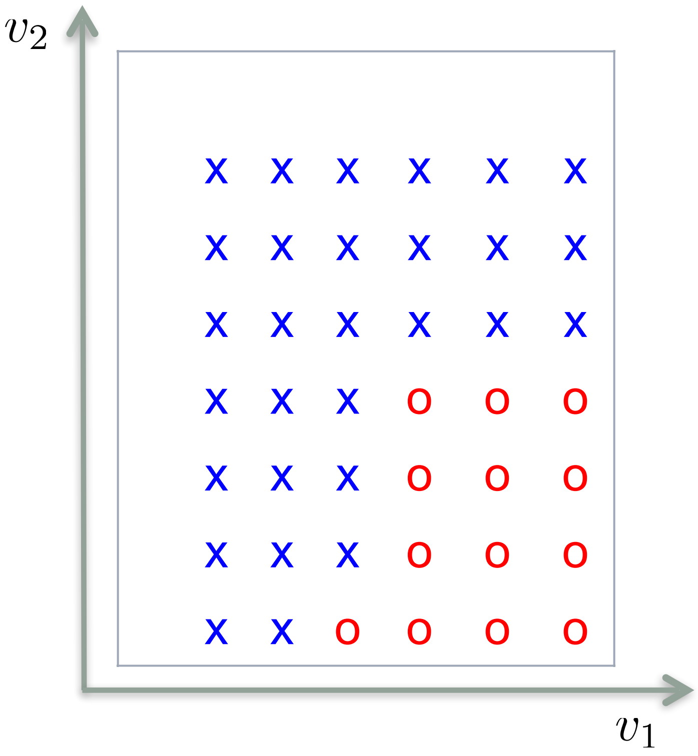


$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi_1(\mathcal{U})}^R \times R_2 \times R_3 \times \ldots \times R_k$$
$$\mathcal{C}_2^{\mathcal{U},R} = \pi_1(\mathcal{U}) \times \overline{\pi_2(\mathcal{U})}^R \times R_3 \times \ldots \times R_k$$
$$\ldots$$
$$\mathcal{C}_k^{\mathcal{U},R} = \pi_1(\mathcal{U}) \times \pi_2(\mathcal{U}) \times \pi_3(\mathcal{U}) \times \ldots \times \overline{\pi_k(\mathcal{U})}^R$$

Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)

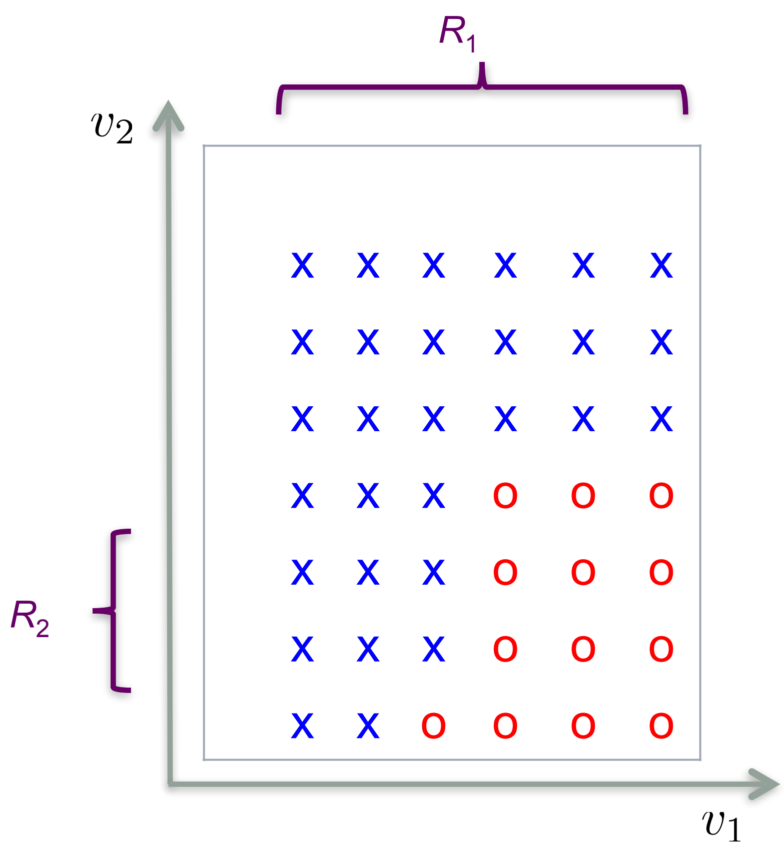


Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



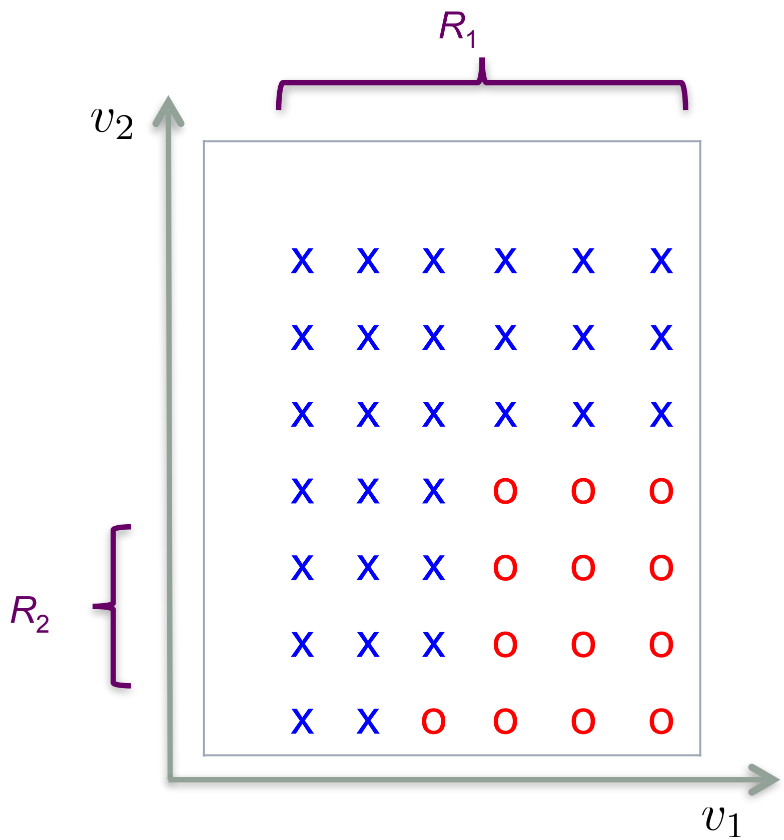
$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^{R} \times R_2 \times R_3 \times \ldots \times R_k$$

Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^{R} \times R_2 \times R_3 \times \ldots \times R_k$$

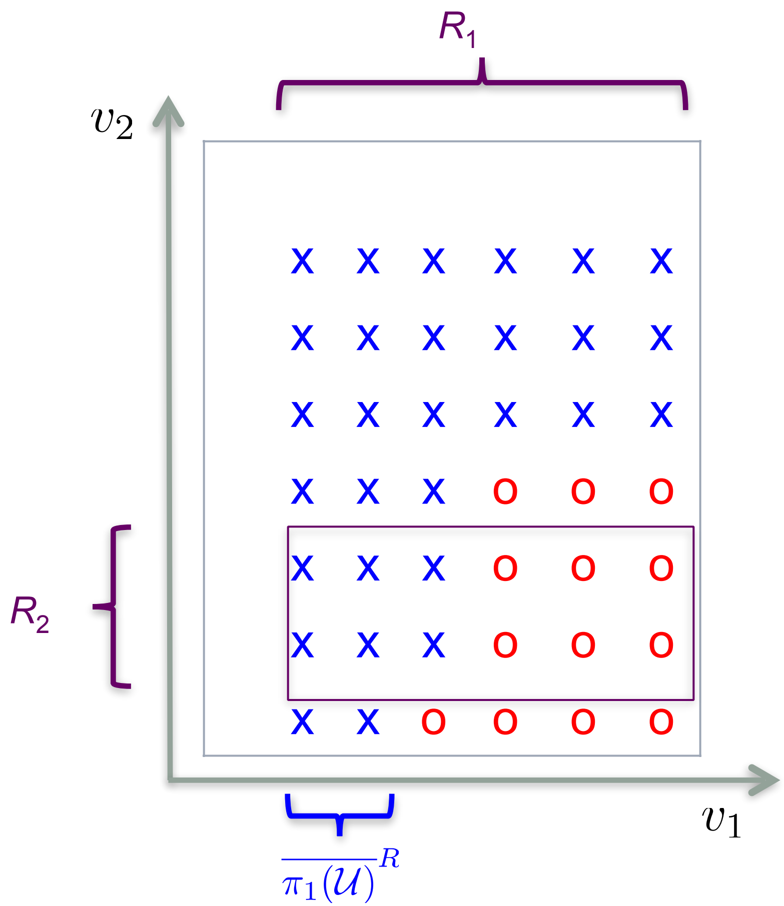
Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^R \times R_2 \times R_3 \times \ldots \times R_k$$

- Lemma 3: If $\overline{\pi_j(\mathcal{U})}^R$ is non-empty no restriction of any other column is possible

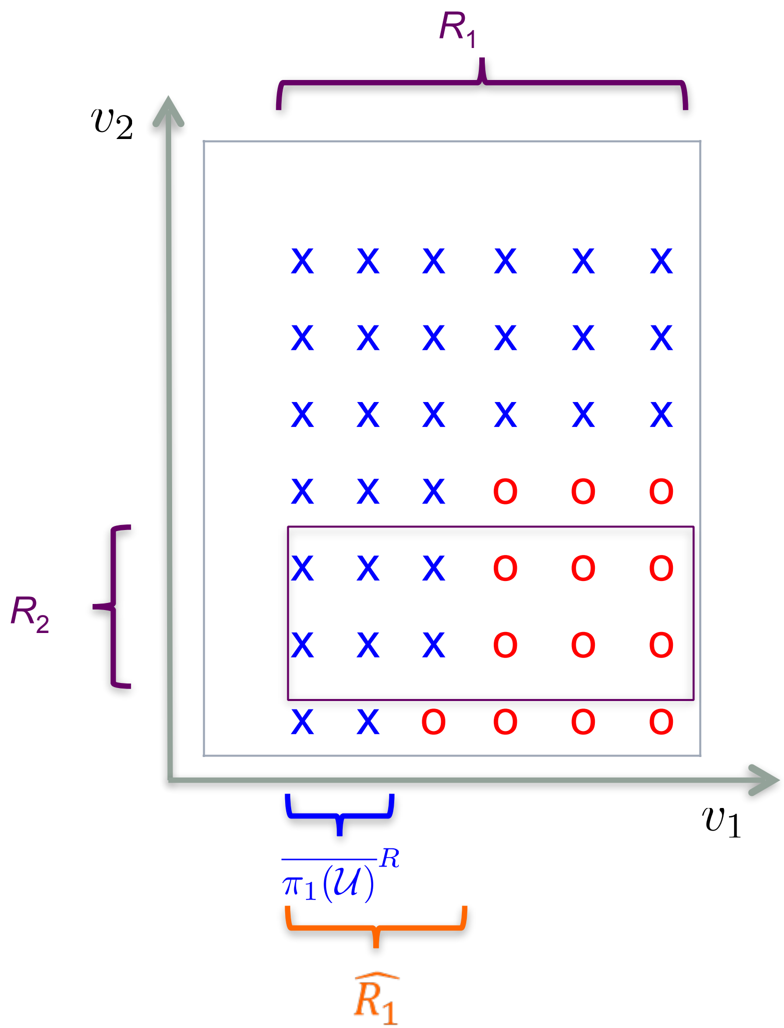
Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^R \times R_2 \times R_3 \times \ldots \times R_k$$

- Lemma 3: If $\overline{\pi_j(\mathcal{U})}^R$ is non-empty no restriction of any other column is possible

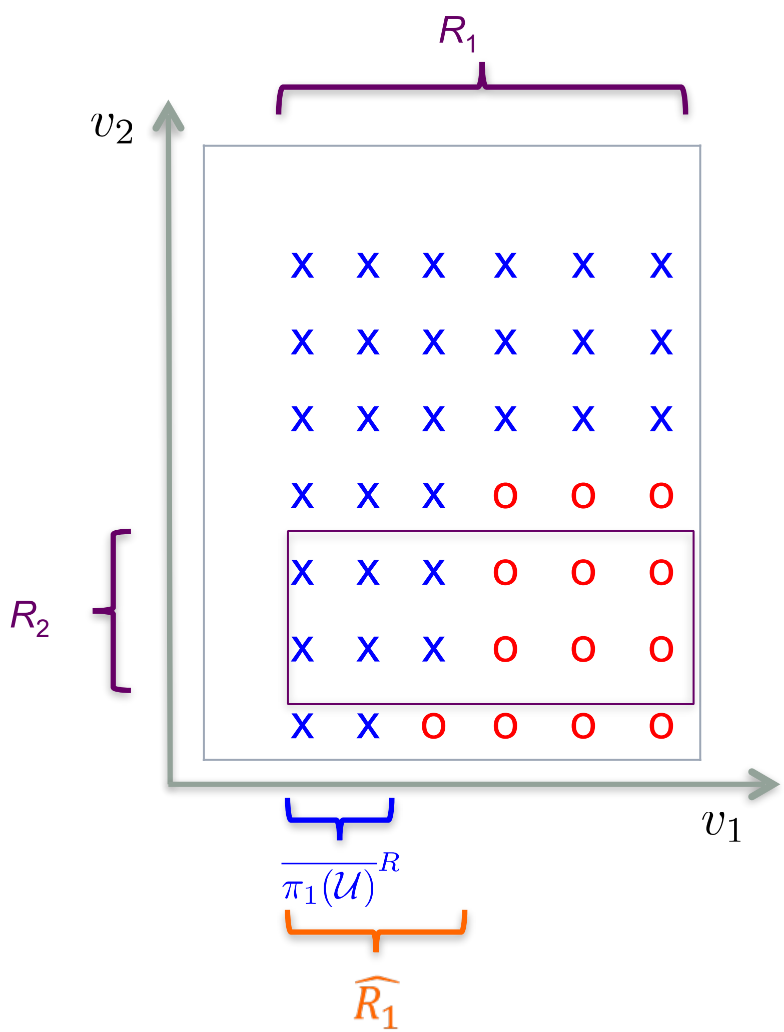
Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^R \times R_2 \times R_3 \times \ldots \times R_k$$

- Lemma 3: If $\overline{\pi_j(\mathcal{U})}^R$ is non-empty no restriction of any other column is possible
  - Here, $R_1$is restricted to $\widehat{R_1}$

Obs#2 – Lemma 3 and its Corollary (Precondition for Filtering)



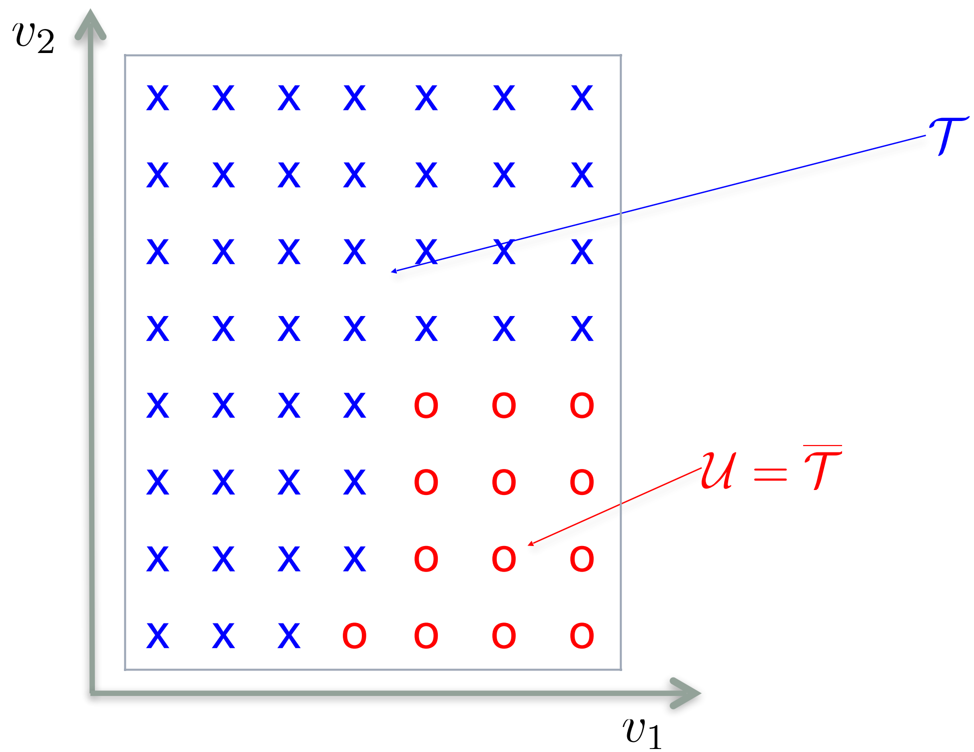
$$\mathcal{C}_1^{\mathcal{U},R} = \overline{\pi(\mathcal{U})_1}^R \times R_2 \times R_3 \times \ldots \times R_k$$

- Lemma 3: If $\overline{\pi_j(\mathcal{U})}^R$ is non-empty no restriction of any other column is possible
  - Here, $R_1$is restricted to $\widehat{R_1}$
- Corollary: If $\overline{\pi_j(\mathcal{U})}^R$ is non-empty for more than one *j*, no restriction via $\mathcal{U}$ is possible at all

Obs#3 – Double Negation

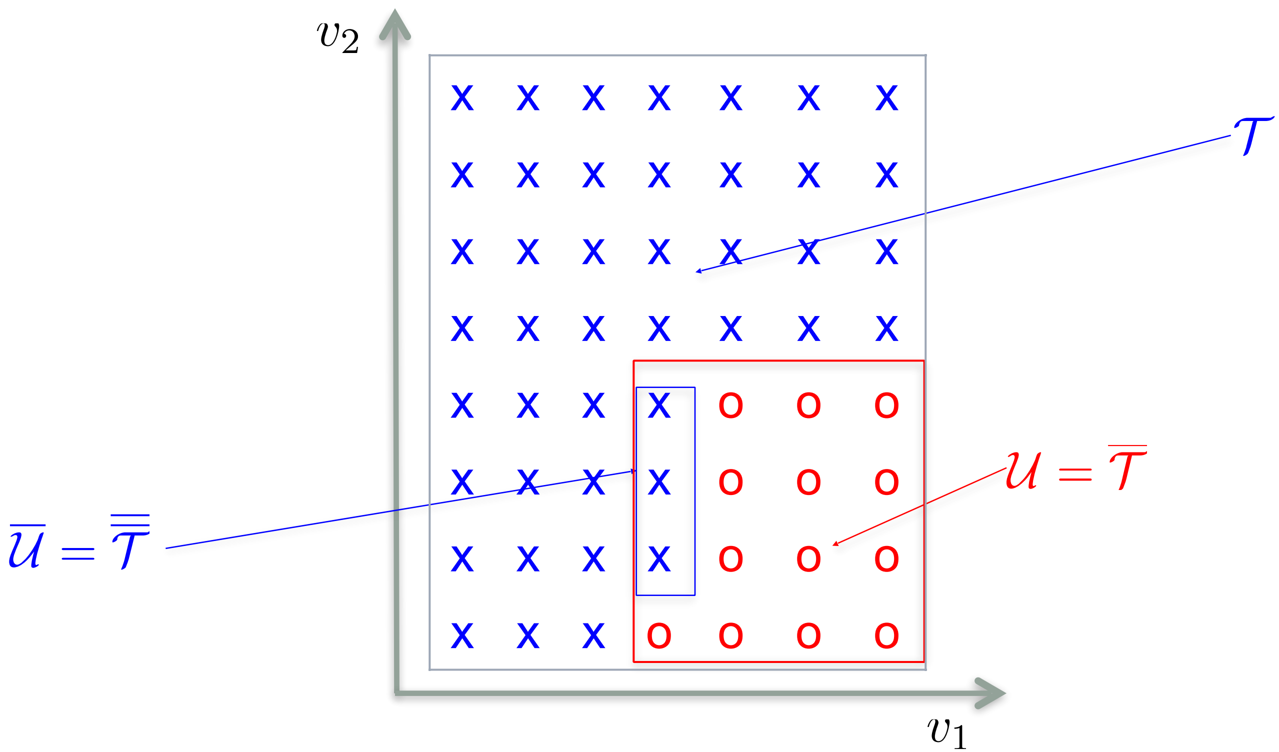
Obs#3 – Double Negation

- Example of t-shirt with extended value domains (denoted here by $\mathcal{T}$ instead of $\mathcal{T}^*$)
- Treat $\mathcal{U} := \overline{\mathcal{T}} := \pi(\mathcal{T}) \setminus \mathcal{T}$ as an NVTAB



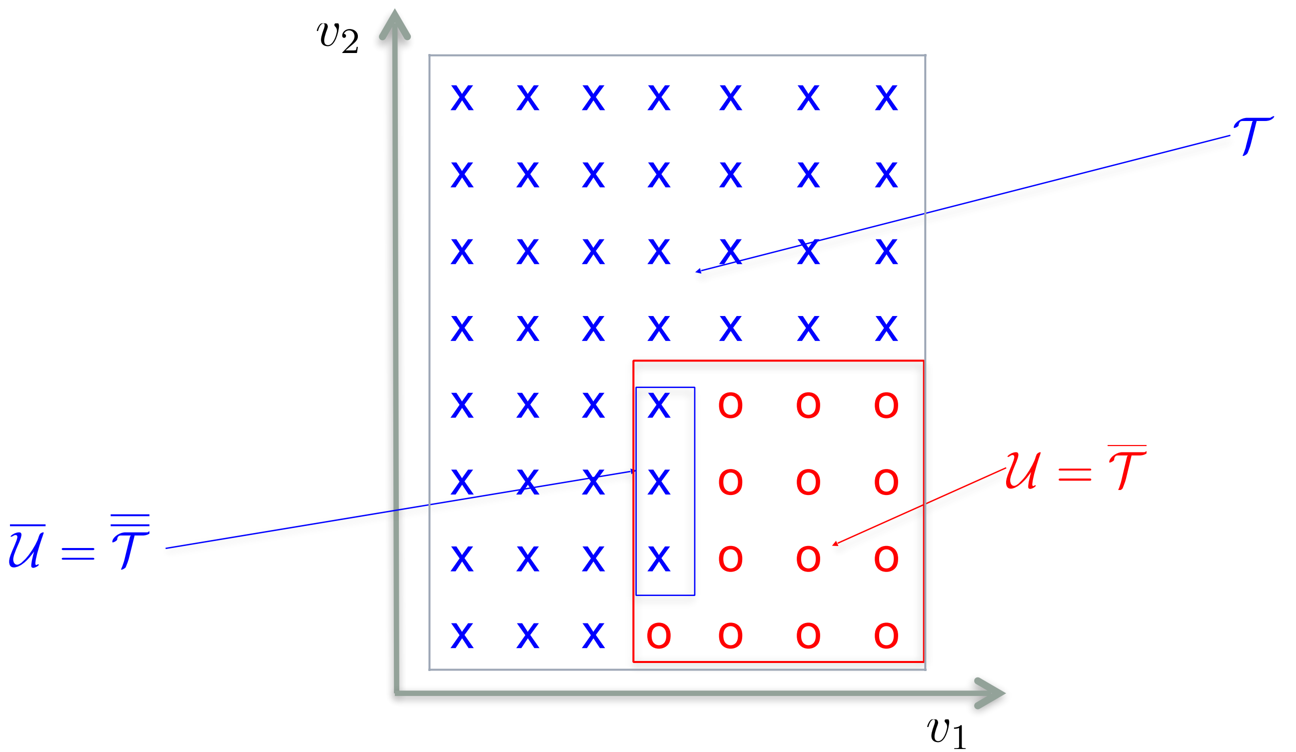
Obs#3 – Double Negation

- Example of t-shirt with extended value domains (denoted here by $\mathcal{T}$ instead of $\mathcal{T}^*$)
- Treat $\mathcal{U} := \overline{\mathcal{T}} := \pi(\mathcal{T}) \setminus \mathcal{T}$ as an NVTAB



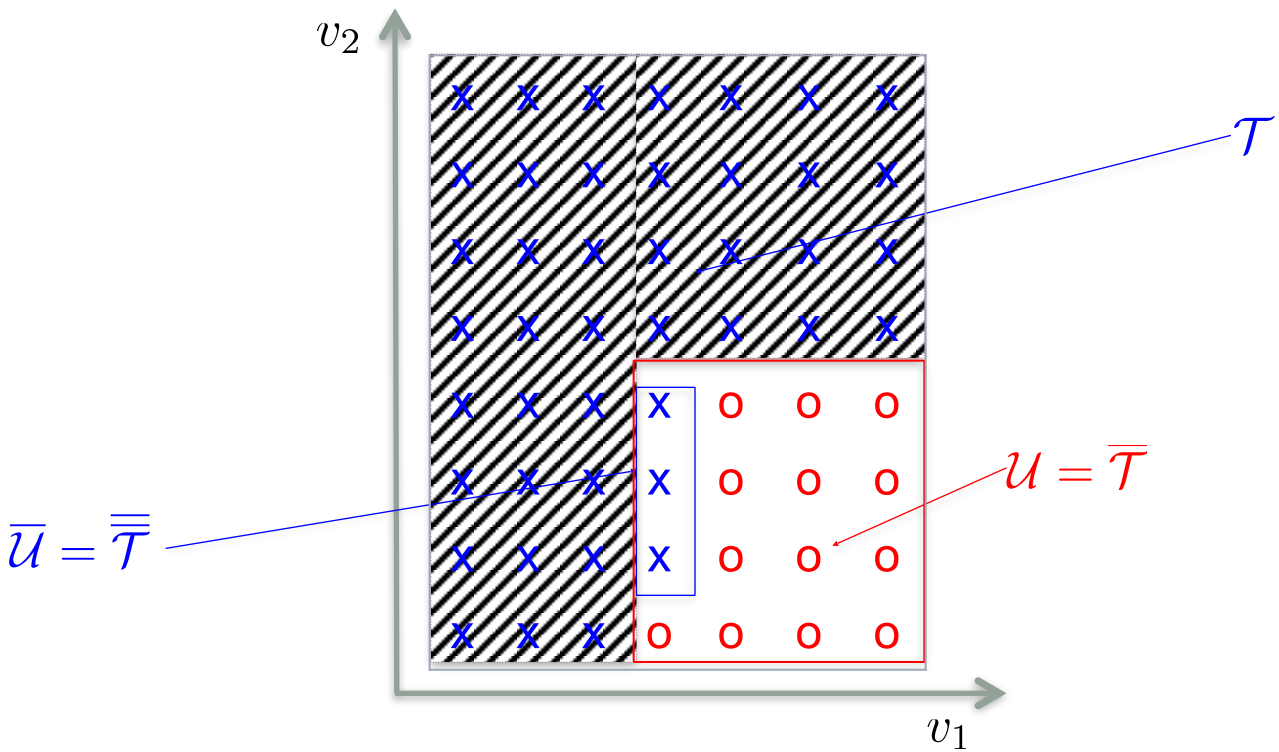
Obs#3 – Double Negation

- Example of t-shirt with extended value domains (denoted here by $\mathcal{T}$ instead of $\mathcal{T}^*$)
- Treat $\mathcal{U} := \overline{\mathcal{T}} := \pi(\mathcal{T}) \setminus \mathcal{T}$ as an NVTAB
- <u>One way of attempting compression of a VTAB</u>



Obs#3 – Double Negation

- Example of t-shirt with extended value domains (denoted here by $\mathcal{T}$ instead of $\mathcal{T}^*$)
- Treat $\mathcal{U} := \overline{\mathcal{T}} := \pi(\mathcal{T}) \setminus \mathcal{T}$ as an NVTAB
- <u>One way of attempting compression of a VTAB</u>



Summary: Features of Proposed Approach

Summary: Features of Proposed Approach

- Existing propagation with positive tables can be used
  - Need ability to complement tables at maintenance-time
  - Need to calculate all $\pi_j(\mathcal{U})$ at maintenance-time
  - Need to calculate all $\overline{\pi_j(\mathcal{U})}^R$ at run-time
    - Stop and exit if two non-empty ones are counted
  - Need to apply arc-consistency on a c-tuple, if one $\pi_j(\mathcal{U})$ is non-empty
  - Apply (pre-existing) constraint propagation algorithm to $\overline{\mathcal{U}}$

Summary: Features of Proposed Approach

- Existing propagation with positive tables can be used
  - Need ability to complement tables at maintenance-time
  - Need to calculate all $\pi_j(\mathcal{U})$ at maintenance-time
  - Need to calculate all $\overline{\pi_j(\mathcal{U})}^R$ at run-time
    - Stop and exit if two non-empty ones are counted
  - Need to apply arc-consistency on a c-tuple, if one $\pi_j(\mathcal{U})$ is non-empty
  - Apply (pre-existing) constraint propagation algorithm to $\overline{\mathcal{U}}$

- Potential performance advantages:
  - Distinction run-time/maintenance-time: smaller $\overline{\mathcal{U}}$
  - Simple test due to Lemma 3/Corollary
  - Double negation one form of potentially compressing tables

Empirical Results

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no “real” test data

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no “real” test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no “real” test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

  - This criterion applied to 57 of them

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no “real” test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

  - This criterion applied to 57 of them
  - Of these 39 were reduced, i.e., $\overline{\overline{\mathcal{T}}}$ compressed had less nodes than $\mathcal{T}$ compressed

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no "real" test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

  - This criterion applied to 57 of them
  - Of these 39 were reduced, i.e., $\overline{\overline{\mathcal{T}}}$ compressed had less nodes than $\mathcal{T}$ compressed
  - Run-time validates correctness of approach

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no “real” test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

  - This criterion applied to 57 of them
  - Of these 39 were reduced, i.e., $\overline{\overline{\mathcal{T}}}$ compressed had less nodes than $\mathcal{T}$ compressed
  - Run-time validates correctness of approach
- Overall, no perceived gains over direct compression with h_2^*

Empirical Results

- VDD implementation (Java) supports NVTABs and double negation
- NVTABs not yet available to customers; hence, no "real" test data
- Double negation was applied to 238 VTABs in VDD test bed if

$$|\mathcal{U}| < |\mathcal{T}|$$

  - This criterion applied to 57 of them
  - Of these 39 were reduced, i.e., $\overline{\overline{\mathcal{T}}}$ compressed had less nodes than $\mathcal{T}$ compressed
  - Run-time validates correctness of approach
- Overall, no perceived gains over direct compression with h_2^*
- Advantage likely in the absence of general compression

Thank You