Conjugacy Class Enumeration in Transformation Semigroups (Conjclassts): User Manual

Part 3: Partial Transformation Semigroups

Clinton Oluranran Kayoh

July 18, 2025

Abstract

This document extends the user manual for the Python package dedicated to enumerating conjugacy classes in transformation semigroups. This continuation focuses specifically on **partial transformation semigroups**, which represent the most general class of transformations by allowing both partial functions and non-injective mappings. The package implements comprehensive algorithms for generating and analyzing these fundamental algebraic structures that encompass both full and partial one-to-one transformations as special cases.

Contents

1 Introduction to Partial Transformation Semigroups

1.1 Mathematical Background

A **partial transformation semigroup** $\mathcal{PT}_n$ on a finite set $X = \{1, 2, \ldots, n\}$ consists of all partial functions from X to X. Unlike partial one-to-one transformations, these functions need not be injective. Transformations are represented as n-tuples where each coordinate can either be an integer (indicating the image) or the symbol '-' (indicating the element is not in the domain).

Definition 1. *The **partial transformation semigroup** $\mathcal{PT}_n$ has size given by:*

$$|\mathcal{PT}_n| = (n+1)^n$$

This counts all functions from an n-element set to an $(n+1)$-element set (including the "undefined" element).

1.2 Relationship to Previous Work

This module generalizes both previous parts:

- Extends Part 1 (Full Transformation Semigroups) by allowing partial functions
- Extends Part 2 (Partial One-to-One Transformation Semigroups) by allowing non-injective mappings
- Provides the most comprehensive framework for transformation semigroups

2 Core Functions for Partial Transformations

2.1 Partial Transformation Semigroup

2.1.1 PT_semigroup(n)

Description: Generates the partial transformation semigroup of degree n by creating all possible n-tuples over the set $\{1, 2, \ldots, n\} \cup \{-\}$.

Mathematical Basis: Implements $\mathcal{PT}_n$, the semigroup of all partial transformations on n points.

Parameters: `n: int` - The degree of the transformation semigroup

Returns: `list` - List of tuples representing partial transformations

Complexity: $\mathcal{O}((n+1)^n)$ time and space complexity

```
                def PT_semigroup(n):
                elements = ["-", *range(1, n+1)]
                combinations = itertools.product(elements, repeat=n)
                combinations_list = list(combinations)
                return combinations_list
```

Example 1. *For $n = 3$, the function returns $4^3 = 64$ transformations: all partial functions from $\{1, 2, 3\}$ to itself, including functions with arbitrary collisions and undefined points.*

2.2 Order-Preserving Partial Transformations

2.2.1 check_P(t)

Description: Checks if a partial transformation preserves order when restricted to its domain.

Mathematical Basis: A partial transformation f is order-preserving if for all x, y in the domain of f with $x \leq y$, we have $f(x) \leq f(y)$.

Parameters: `t: tuple` - A partial transformation

Returns: `bool` - True if the transformation is order-preserving

```
                def check_P(t):
                if all(isinstance(x, str) for x in t):
                return True
                if isinstance(t[0], int) and all(isinstance(x, str) for x in t[1:]) or \
                isinstance(t[-1], int) and all(isinstance(x, str) for x in t[:-1]):
                return True
                if all(isinstance(x, int) for x in t):
                if sorted(t) == list(t) or sorted(t, reverse=False) == list(t):
                return True
                if any(isinstance(x, int) for x in t) and any(isinstance(x, str) for x in t):
                int_indices = [i for i, x in enumerate(t) if isinstance(x, int)]
                if sorted(int_indices) == int_indices:
                int_values = [t[i] for i in int_indices]
                if sorted(int_values) == int_values or sorted(int_values, reverse=False) ==
                    int_values:
                return True
                return False
```

2.2.2 Order_Preserving_PT(tuples)

Description: Filters partial transformations to those that preserve order.

Parameters: `tuples: list` - List of partial transformations from PT_semigroup

Returns: `list` - List of order-preserving partial transformations

2.3 Order-Preserving or Reversing Partial Transformations

2.3.1 check_PR(t)

Description: Checks if a partial transformation is either order-preserving or order-reversing.

Mathematical Basis: Determines if the defined points in the transformation form a monotone sequence (either non-decreasing or non-increasing).

Parameters: `t: tuple` - A partial transformation

Returns: `bool` - True if the transformation is monotone

2.3.2 Order_Preserving_Or_Reversing_PT(tuples)

Description: Selects partial transformations that are either entirely order-preserving or entirely order-reversing.

Parameters: `tuples: list` - List of partial transformations

Returns: `list` - List of monotone partial transformations

2.4 Order-Decreasing Partial Transformations

2.4.1 check_decreasing(t)

Description: Checks if a partial transformation satisfies the order-decreasing condition: for each position i, if the value is an integer, it must be $\leq i + 1$.

Mathematical Basis: Implements the condition that transformations map elements to positions not exceeding their original order.

Parameters: `t: tuple` - A partial transformation

Returns: `bool` - True if the transformation is order-decreasing

```
def check_decreasing(t):
for i, x in enumerate(t):
if not isinstance(x, str) and isinstance(x, int) and x > i + 1:
return False
return True
```

2.4.2 Order_Decreasing_PT(tuples)

Description: Filters partial transformations to those satisfying the order-decreasing condition.

Parameters: `tuples: list` - List of partial transformations

Returns: `list` - List of order-decreasing partial transformations

3 Contraction Partial Transformation Semigroups

3.1 Partial Contraction Transformations

3.1.1 contraction_check(t)

Description: Checks if a partial transformation is a contraction, meaning that adjacent defined points map to values differing by at most 1.

Mathematical Basis: A partial transformation f is a contraction if for all adjacent defined points i, j in the domain, $|f(i) - f(j)| \leq 1$.

Parameters: `t: tuple` - A partial transformation

Returns: `bool` - True if the transformation is a contraction

```
def contraction_check(t):
if all(isinstance(x, str) for x in t):
return True
if not any(isinstance(x, int) for x in t):
return False
for i in range(len(t) - 1):
if isinstance(t[i], int) and isinstance(t[i + 1], int):
if abs(t[i] - t[i + 1]) > 1:
return False
return True
```

3.1.2 Contraction_PT(tuples)

Description: Generates the partial contraction transformation semigroup.

Parameters: `tuples: list` - List of partial transformations

Returns: `list` - List of contraction partial transformations

3.2 Combined Property Semigroups

3.2.1 Order-Preserving Contraction PT

Description: Intersection of order-preserving and contraction partial transformations.

Usage: `Order_Preserving_PT(Contraction_PT(combinations))`

3.2.2 Order-Preserving or Reversing Contraction PT

Description: Intersection of monotone and contraction partial transformations.

Usage: `Order_Preserving_Or_Reversing_PT(Contraction_PT(combinations))`

3.2.3 Order-Decreasing Contraction PT

Description: Intersection of order-decreasing and contraction partial transformations.

Usage: `Order_Decreasing_PT(Contraction_PT(combinations))`

4 Conjugacy Classes for Partial Transformations

4.1 Graph Representation

For partial transformations, the functional graph representation handles both undefined points and arbitrary mappings:

- Nodes represent elements of $\{1, 2, \ldots, n\}$
- Edges $i \to j$ exist when the transformation maps i to j
- No edge from i indicates i is not in the domain
- Multiple edges to the same target indicate non-injective mappings
- Conjugacy preserves both the undefined pattern and the mapping structure

4.1.1 conjugacy_class(tuples_list)

Description: Partitions partial transformations into conjugacy classes using graph isomorphism.

Mathematical Basis: Two partial transformations are conjugate if there exists a permutation that preserves both the domain/range structure and the functional dependencies.

Note: The same function works across all three semigroup types due to the unified graph representation.

5 Usage Examples

5.1 Basic Partial Semigroup Generation

```
# Generate partial transformation semigroup for n=3
PT3 = PT_semigroup(3)
print(f"Partial transformation semigroup size: {len(PT3)}")

# Generate order-preserving partial transformations
OP_PT3 = Order_Preserving_PT(PT3)
print(f"Order-preserving partial transformations: {len(OP_PT3)}")

# Generate contraction partial transformations
Cont_PT3 = Contraction_PT(PT3)
print(f"Contraction partial transformations: {len(Cont_PT3)}")
```

5.2 Combined Property Semigroups

```
# Generate order-preserving contraction partial transformations
OP_Cont_PT3 = Order_Preserving_PT(Contraction_PT(PT3))
print(f"Order-preserving contraction partial: {len(OP_Cont_PT3)}")

# Conjugacy class analysis for specialized semigroups
classes = conjugacy_class(OP_Cont_PT3)
for class_id, elements in classes.items():
print(f"Class {class_id}: {len(elements)-1} elements")
```

6 Mathematical Properties

Theorem 1. *The partial transformation semigroup $\mathcal{PT}_n$ contains both the full transformation semigroup $\mathcal{T}_n$ and the symmetric inverse semigroup $\mathcal{I}_n$ as subsemigroups:*

$$\mathcal{T}_n \subset \mathcal{PT}_n \quad \text{and} \quad \mathcal{I}_n \subset \mathcal{PT}_n$$

Theorem 2. *For partial transformations, conjugacy classes correspond to isomorphism types of what may be called "partial functional multigraphs," where isomorphism preserves:*

- *The pattern of undefined domain elements*
- *The functional dependency structure*
- *The collision patterns for non-injective mappings*

Theorem 3. *The number of conjugacy classes in $\mathcal{PT}_n$ equals the number of isomorphism types of directed graphs on n vertices where each vertex has out-degree at most 1 (allowing multiple edges to the same target).*

7 Comparison of All Three Semigroup Types

Property	**Full $\mathcal{T}_n$**	**Partial 1-1 $\mathcal{I}_n$**	**Partial $\mathcal{PT}_n$**
Size	n^n	$\sum_{k=0}^{n} \binom{n}{k}^2 k!$	$(n+1)^n$
Injective	No	Yes	No
Surjective	No	Partial	Partial
Domain	Full set	Subset	Subset
Structure	Semigroup	Inverse semigroup	Semigroup
Complexity	Medium	High	Highest

Table 1: Comparison of all three transformation semigroup types

8 Computational Complexity Analysis

8.1 Size Growth

n	$\lvert\mathcal{T}_n\rvert$	$\lvert\mathcal{I}_n\rvert$	$\lvert\mathcal{PT}_n\rvert$
1	1	2	2
2	4	7	9
3	27	34	64
4	256	209	625
5	3125	1546	7776
6	46656	13327	117649

Table 2: Exponential growth of semigroup sizes

8.2 Practical Limitations

- $\mathcal{T}_n$: Practical for $n \leq 7$
- $\mathcal{I}_n$: Practical for $n \leq 6$
- $\mathcal{PT}_n$: Practical for $n \leq 5$

9 Limitations and Future Development

9.1 Current Limitations

- Exponential complexity severely limits practical computation
- Memory requirements grow as $(n+1)^n$ for partial transformations
- Graph isomorphism becomes bottleneck for larger n
- No specialized algorithms exploiting semigroup structure

9.2 Planned Features

TODO:

- Efficient sparse representation for partial transformations
- Canonical labeling for functional graphs
- Exploitation of Green's relations for conjugacy classification
- Parallel computation for larger instances
- Algebraic properties and structure theory integration

10 Conclusion

This final part completes the comprehensive framework for transformation semigroup analysis, providing tools for the most general class of partial transformations. The implementation handles the combinatorial complexity of arbitrary partial functions while maintaining mathematical consistency across all three semigroup types.

The package now offers a unified approach to studying transformation semigroups, from the restricted case of full transformations through the intermediate case of partial one-to-one transformations to the most general case of arbitrary partial transformations.

References

[1] Howie, J. M. (1995). *Fundamentals of Semigroup Theory*. Oxford University Press.

[2] Lipscomb, S. (1996). *Symmetric Inverse Semigroups*. American Mathematical Society.

[3] Schein, B. M. (1970). *Relations and transformations*. Semigroup Forum.

[4] Ganyushkin, O., & Mazorchuk, V. (2008). *Classical Finite Transformation Semigroups*. Springer.

[5] Abusarris, H., & Ayık, G. (2023). On the rank of generalized order-preserving transformation semigroups. *Turkish Journal of Mathematics*. https://doi.org/10.55730/1300-0098.3420.

[6] Adan-Bante, E. (2005). On nilpotent groups and conjugacy classes. *arXiv: Group Theory*, 345-356.

[7] Ahmad, A., Magidin, A., & Morse, R. (2012). Two generator p-groups of nilpotency class 2 and their conjugacy classes. *Publicationes Mathematicae Debrecen*, 81, 145-166. https://doi.org/10.5486/PMD.2012.5114.

[8] Akinwunmi, S.A. & Makanjuola, S.O. (2019). Enumeration Partial Contraction Transformation Semi-groups. *Journal of the Nigerian Association of Mathematical Physics*, 29(B), 70-77.

[9] Akinwunmi, S.A., Mogbonju, M.M., Adeniji, A.O., Oyewola, D.O., Yakubu, G., Ibrahim, G.R., & Fatai, M.O. (2021). Nildempotency Structure of Partial One-One Contraction CIn Transformation Semigroups. *International Journal of Research and Scientific Innovation (IJRSI)*, 8(1), 230-236.

[10] Mogbonju, M.M., Gwary, T.M., & Ojeniyi, A.B. (2014). Presentation of Conjugacy Classes in the Partial Order-Preserving Transformation Semigroup with the aid of graph. *Mathematical Theory and Modeling*, 4, 110-142.

[11] Mohammadian, A., & Erfanian, A. (2018). On the nilpotent conjugacy class graph of groups. *Journal of Algebra and Its Applications*, 37, 77-90. https://doi.org/10.1142/S0219498818500407.

[12] Ugbene, I.J., Eze, E.O., & Makanjuola, S.O. (2013). On the Number of Conjugacy Classes in the Injective Order-Decreasing Transformation Semigroup. *Pacific Journal of Science and Technology*, 14(1), 182-186.

[13] Ugbene, I.J. & Makanjuola, S.O. (2012). On the Number of Conjugacy Classes in the Injective Order-preserving Transformation Semigroup. *Icastor Journal of Mathematical Sciences*, 6(1), 1-8.

[14] Ugbene, I.J., Suraju, O., & Ugochukwu, N. (2021). Digraph of the full transformation semigroup. *Journal of Discrete Mathematical Sciences and Cryptography*, 25(8), 2457-2465. https://doi.org/10.1080/09720529.2020.1862955.