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

Part 2: Partial One-to-One Transformation Semigroups

Clinton Oluranran Kayoh

July 05, 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 one-to-one transformation semigroups**, which generalize full transformation semigroups by allowing partial functions and injective restrictions. The package implements specialized algorithms for generating and analyzing these important algebraic structures.

Contents

1 Introduction to Partial One-to-One Transformation Semigroups

1.1 Mathematical Background

A **partial one-to-one transformation semigroup** $\mathcal{I}_n$ on a finite set $X = \{1, 2, \ldots, n\}$ consists of all partial injective functions from X to X. These 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 one-to-one transformation semigroup** $\mathcal{I}_n$ has size given by:*

$$|\mathcal{I}_n| = \sum_{k=0}^{n} \binom{n}{k}^2 k!$$

where k ranges over possible domain sizes.

1.2 Relationship to Previous Work

This module extends the functionality described in Part 1 (Full Transformation Semigroups) by:

- Handling partial functions with undefined points
- Enforcing injectivity constraints
- Providing specialized order-preserving variants for partial transformations

2 Core Functions for Partial One-to-One Transformations

2.1 Partial One-to-One Transformation Semigroup

2.1.1 P1T_semigroup(n)

Description: Generates the partial one-to-one transformation semigroup of degree n by creating all valid n-tuples over the set $\{1, 2, \ldots, n\} \cup \{-\}$ that satisfy the injectivity condition.

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

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

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

Complexity: $\mathcal{O}(n! \cdot 2^n)$ due to injectivity constraint

```
def P1T_semigroup(n):
elements = ["-", *range(1, n+1)]
combinations = itertools.product(elements, repeat=n)
unique_combinations = filter(lambda x: all(x.count(i) <= 1
for i in range(1, n+1)), combinations)
unique_combinations_list = list(unique_combinations)
return unique_combinations_list
```

Example 1. *For $n = 2$, the function returns 7 transformations: all partial injective functions from $\{1, 2\}$ to itself, including the empty function.*

2.2 Order-Preserving Partial Transformations

2.2.1 Order_Preserving_P1T(combinations)

Description: Filters partial one-to-one transformations to those that preserve order when restricted to their 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: `combinations: list` - List of partial transformations from P1T_semigroup

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

```
def Order_Preserving_P1T(combinations):
selected_combinations = []
for combo in combinations:
has_int = False
has_str = False
last_int = None
for item in combo:
if isinstance(item, int):
has_int = True
if last_int is None:
last_int = item
elif item <= last_int:
break
else:
last_int = item
elif isinstance(item, str):
has_str = True
else:
break
else:
if has_int and has_str:
selected_combinations.append(combo)
elif not has_int:
selected_combinations.append(combo)
elif has_int and not has_str and list(combo) == sorted(list(combo)):
selected_combinations.append(combo)
return selected_combinations
```

2.3 Order-Preserving or Reversing Partial Transformations

2.3.1 check_tuple(t)

Description: Helper function that checks if a partial transformation is either order-preserving or order-reversing.

Mathematical Basis: Determines if the integer elements 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_P1T(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_P1T(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 One-to-One 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_P1T(tuples)

Description: Generates the partial one-to-one 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 P1T

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

Usage: `Order_Preserving_P1T(Contraction_P1T(combinations))`

3.2.2 Order-Preserving or Reversing Contraction P1T

Description: Intersection of monotone and contraction partial transformations.

Usage: `Order_Preserving_Or_Reversing_P1T(Contraction_P1T(combinations))`

3.2.3 Order-Decreasing Contraction P1T

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

Usage: `Order_Decreasing_P1T(Contraction_P1T(combinations))`

4 Conjugacy Classes for Partial Transformations

4.1 Graph Representation

For partial transformations, the functional graph representation is extended to handle undefined points:

- 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
- The conjugacy relation remains: two partial transformations are conjugate iff their functional graphs are isomorphic

4.1.1 conjugacy_class(tuples_list)

Description: Partitions partial transformations into conjugacy classes using graph isomorphism, adapted for partial functions.

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

Note: The same function from Part 1 works for partial transformations due to the flexible graph representation.

5 Usage Examples

5.1 Basic Partial Semigroup Generation

```python
# Generate partial one-to-one transformation semigroup for n=3
I3 = P1T_semigroup(3)
print(f"Partial one-to-one semigroup size: {len(I3)}")

# Generate order-preserving partial transformations
OP_I3 = Order_Preserving_P1T(I3)
print(f"Order-preserving partial transformations: {len(OP_I3)}")

# Generate contraction partial transformations
Cont_I3 = Contraction_P1T(I3)
print(f"Contraction partial transformations: {len(Cont_I3)}")
```

5.2 Combined Property Semigroups

```
# Generate order-preserving contraction partial transformations
OP_Cont_I3 = Order_Preserving_P1T(Contraction_P1T(I3))
print(f"Order-preserving contraction partial: {len(OP_Cont_I3)}")

# Conjugacy class analysis for specialized semigroups
classes = conjugacy_class(OP_Cont_I3)
for class_id, elements in classes.items():
print(f"Class {class_id}: {len(elements)-1} elements")
```

6 Mathematical Properties

Theorem 1. *For partial one-to-one transformations, conjugacy classes correspond to isomorphism types of partial functional graphs, where isomorphism preserves both the edge structure and the pattern of undefined points.*

Theorem 2. *The number of conjugacy classes in $\mathcal{I}_n$ equals the number of isomorphism types of directed graphs on n vertices where each vertex has out-degree at most 1.*

7 Comparison with Full Transformation Semigroups

Property	**Full $\mathcal{T}_n$**	**Partial $\mathcal{I}_n$**
Size	n^n	$\sum_{k=0}^{n} \binom{n}{k}^2 k!$
Injective	No	Yes
Surjective	No	Partial
Domain	Full set	Subset
Conjugacy	Graph isomorphism	Extended graph isomorphism

Table 1: Comparison of full and partial transformation semigroups

8 Limitations and Future Development

8.1 Current Limitations

- Exponential complexity limits practical computation to $n \leq 6$
- Basic graph isomorphism without optimization for partial functions
- No specialized algorithms for inverse semigroup structure

8.2 Planned Features

TODO:

- Efficient representation using sparse matrices
- Exploitation of inverse semigroup properties
- Specialized conjugacy algorithms for partial transformations
- Integration with algebraic theory of inverse semigroups

9 Conclusion

This extension to the transformation semigroup package provides comprehensive tools for working with partial one-to-one transformations, an important class in semigroup theory with applications in computer science and combinatorics. The implementation handles the combinatorial complexity of partial functions while maintaining mathematical rigor.

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] 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.

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.