SoftwareX 24 (2023) 101554

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

ParShift: a Python package to study order and differentiation in group conversations

Bruno D. Ferreira-Saraiva [a,b], João P. Matos-Carvalho [a,b], Nuno Fachada [a], Manuel Pita [b,a,*]

[a] COPELABS, Universidade Lusófona, Campo Grande 376, 1749-024 Lisboa, Portugal
[b] CICANT, Universidade Lusófona, Campo Grande 376, 1749-024 Lisboa, Portugal

ARTICLE INFO

Keywords:
Small groups
Social interaction
Participation shifts
Interpersonal coordination
Turn-taking
Emergent social behaviour

ABSTRACT

Collective organization in multi-party conversations emerges through the exchange of utterances between participants. While most research has focused on content-centred mechanisms that lead to emergent conversational coordination, less attention has been given to explaining conversational order based on who is addressed and who responds, especially when dealing with large conversational datasets. In this paper, we introduce a Python library, *ParShift*, that implements a state-of-the-art theoretical quantitative framework known as *Participation Shifts*. This framework enables researchers to study participant-centred order and differentiation in multi-party conversations. With *ParShift*, researchers can characterize conversations by quantifying the probabilities of events related to how people address each other during conversations. This library is particularly useful for studying conversation threads in social networks, parliamentary debates, team meetings, or student debates on a large scale.

Code metadata

Current code version	v1.0.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00407
Code Ocean compute capsule	Not applicable
Legal Code License	MIT
Code versioning system used	Git
Software code languages, tools, and services used	Python
Compilation requirements, operating environments & dependencies	*Python* $\geq$ 3.8, *pandas* $\geq$ 1.5.2, *matplotlib* $\geq$ 3.6.2, *squarify* $\geq$ 0.4.3
If available Link to developer documentation/manual	https://bdfsaraiva.github.io/parshift
Support email for questions	bruno.saraiva@ulusofona.pt

1. Introduction

1.1. Motivation and significance

The rapid evolution of the hyper-connected information ecosystem on the Internet has created new scenarios in which the study of multi-party conversation is increasingly important because of its potential societal impact on many scales. Indeed, group conversations have been widely studied in the face-to-face context like classrooms, organizations and other social gatherings [1]. On the other hand, mediated conversations that take place online, or through technological communication devices, have specific constraints and interactional affordances that not only influence the ways in which collective dynamics unfold, but also require new research approaches, as well as novel computational methods, to make sense of large volumes of conversational data [2].

In their seminal work, Sacks et al. approached conversation as a system in which interactional rules drive the emergence of conversational order [3]. One aspect of conversational order is linked to the interactional rules that allow actors to align with, or differentiate from, other actors. Alignment and differentiation rules are enacted through linguistic entrainment, use of stance expressions, pauses in communication, and topic management—all of this influenced by factors such as

* Corresponding author at: CICANT, Universidade Lusófona, Campo Grande 376, 1749-024 Lisboa, Portugal.
E-mail address: manuel.pita@ulusofona.pt (Manuel Pita).

https://doi.org/10.1016/j.softx.2023.101554
Received 29 June 2023; Received in revised form 13 September 2023; Accepted 6 October 2023

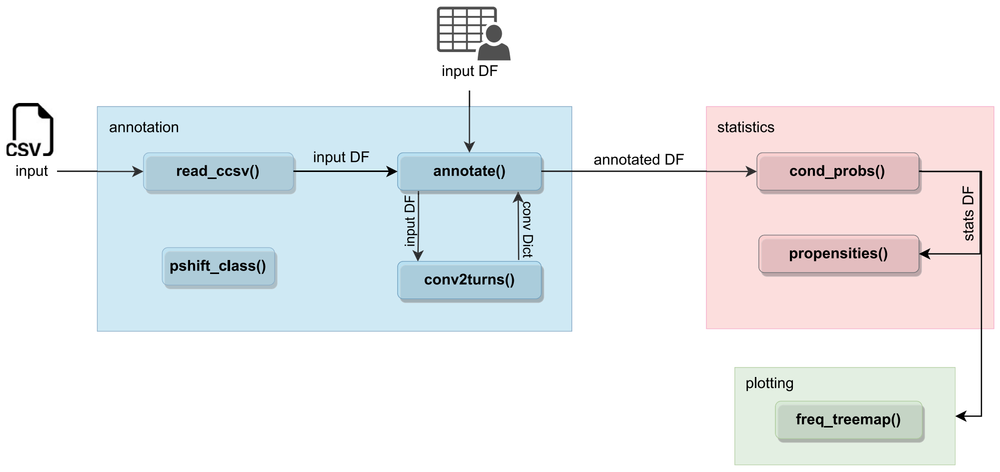

Fig. 1. Architecture of the *ParShift* package. The three core modules are represented as boxes, with their respective functions shown within rounded rectangles. The arrows represent the data flow in the package's computational pipeline ("DF" stands for a Pandas data frame).

power dynamics, individual preferences, as well as social and cultural norms.

Group conversations often unfold cyclically, with participants taking turns, producing, and consuming utterances. Researchers have identified various stages or phases common to many conversations. Tuckman et al. [4,5] proposed a widely accepted four-stage model of conversation that considers, (a) forming, (b) storming, (c) norming, and (d) performing. In the *forming* stage, participants get to know each other and establish a sense of cohesion, typically by alignment of their language. In the *storming* stage, conflicts may arise as group members assert themselves and express diverging opinions through differentiating interactions. The *norming* stage involves resolving these conflicts and establishing shared expectations for behaviour. Finally, group members work together towards their goals in the *performing* stage.

The study of alignment and differentiation in group conversations has practical applications in education, business, political participation, collective identity and conflict resolution. For example, understanding the emergence of conversational order in classroom discussions can inform the design of instructional strategies that promote student engagement and interaction. In business, the study of order and differentiation can provide insights into how to facilitate effective team meetings and negotiations.

The Python library we introduce here, *ParShift*, is based on a framework for analysing conversation—at the level of adjacent turn pairs—that incorporates both alignment and differentiation, grounded on the notion of *participation shift* [6]. A participation shift refers to the constant change in individuals' roles as speaker, target, or unaddressed recipient. This framework connects the study of sequential talk production with interest in conversational discrimination in small-group research and highlights new avenues for further micro-sociological investigation.

ParShift enables researchers to study how people engage in group conversations in terms of the collective patterns resulting from the ways in which they address each other. Gibson's theorethical framework allows us to approach the unfolding of a group conversation as a dynamical system, where interactions between interlocutors at the micro level drive emergent order at the macro level. Here the focus is on how interlocutors become linked to others through explicit addressing and replying behaviours.

Social and communication scientists can use *ParShift* for a wide range of research questions including, (a) the study of gender conversational dynamics based on who gets targeted, who responds and who usurps the floor in conversations about different subjects in large text corpora (b) analysing the transcripts of work meetings, providing valuable insights into how people work together in groups, including how they coordinate their actions, allocate tasks, manage information flow, and develop friendships (c) the study of political debates, e.g. in parliaments over long periods, to understand the evolution of different multi-party political systems, or (d) to examine student group debates to understand participation and engagement, thus providing evidence to inform educational interventions that improve dialogue [7–10].

Table 1
The thirteen specific participation shifts identified by Gibson [6], grouped by p-shift class (leftmost column).

P-shifts	Example	
Turn Receiving	AB-BA	John talks to Mary, then Mary replies.
	AB-B0	John talks to Mary, then Mary addresses the group.
	AB-BY	John talks to Mary, then Mary talks to Irene.
Turn Claiming	A0-X0	John talks to the group, then Frank talks to the group.
	A0-XA	John talks to the group, then Frank talks to John.
	A0-XY	John talks to the group, then Frank talks to Mary.
Turn Usurping	AB-X0	John talks to Mary, then Frank talks to the group.
	AB-XA	John talks to Mary, then Frank talks to John.
	AB-XB	John talks to Mary, then Frank addresses Mary.
	AB-XY	John talks to Mary, then Frank addresses Irene.
Turn Continuing	A0-AY	John talks to the group, then addresses Mary.
	AB-A0	John talks to Mary, then makes a remark to the group.
	AB-AY	John talks to Mary then to Irene.

1.2. Background concepts

The notion of participation shift is at the core of the framework proposed by Gibson [6] to analyse small-group conversational dynamics from an actor-centred perspective. This theoretical framework allows researchers to understand the mechanisms that drive the development of collective behaviours and the factors that influence their formation and evolution over time. Gibson defined a participation shift as a labelled pair of adjacent turns in a multi-party conversation, henceforth p-shift, T_i-T_j, where T_i represents who speaks and who is addressed, and T_j who responds and who the second speaker targets. It is important to emphasize that Gibson's framework is concerned with the *explicit* targeting of speakers, by addressing or replying, and not with possible implicit targeting that may be inferred from the conversation's content. See [11] for work relevant to the problem of target inference.

The first turn, T_i, always starts with the character A, representing the current (known) speaker, who may target an interlocutor specifically, B, or the group, 0. The second turn, T_j, constitutes the actual shift: (a) A may continue their turn (but changing target); (b) the targeted interlocutor, B, replies; (c) a different interlocutor, X, speaks after A addressed the group; or, (d) an interlocutor who was not the target in the first turn, X, takes the floor. For each type of second turn in the shift, the new speaker may target a specific interlocutor or speak to the group. These combinations produce the thirteen p-shifts, each belonging to one of the four p-shift classes depicted in Table 1.

For a given conversation, the Participation Shifts framework counts the raw frequencies of each p-shift label. From these, it becomes possible to compute the conditional probabilities of p-shifts, given they were directed (or undirected). Further conditional probabilities are computed by considering whether there was a change of speaker, effectively separating turn continuation by, A, from the other twelve p-shifts. Gibson synthesizes the conditional probabilities as three *propensities*. The first is the turn-receiving propensity, which represents the likelihood of the target in one turn becoming the speaker in the next (corresponding to AB-BA, AB-$B0$, and AB-BY). The second propensity is the targeting propensity, which denotes the probability of an unaddressed recipient in one turn being addressed in the next. Finally, the termination propensity quantifies the probability that the group allowed speakers to retain the floor.

Recent works that incorporate Gibson's theoretical framework include (a) investigations into the turn-taking patterns of fourth-grade students within group settings in [12] and (b) explorations into the specific phenomenon of turn-usurping in dialogic collaborative problem solving [13]. Both studies leverage Gibson's framework to gain a comprehensive understanding of how students engage in dialogue and access the conversational floor, shedding light on the dynamics of participation in collaborative learning environments. A large-scale corpus released for the task of conversation disentanglement [14] could be analysed with Gibson's framework, using the package we introduce here.

2. Software description

2.1. Software architecture

ParShift is written in Python and provides functions for reading group conversations, computing the quantitative variables that are part of the p-shifts framework described in Section 1, and producing tables and graphical representations of these variables. The *ParShift* code depends on standard and well-supported Python libraries, mainly Pandas [15] and Matplotlib [16]. The package contains four modules. Three of them, `annotation`, `statistics` and `plotting`, are detailed in Fig. 1. The fourth module, `oo_parshift`, provides an object-oriented interface for the previous modules with the goal of simplifying the package's high-level usability.

The `annotation` module contains the functions that allow *ParShift* to read input files and annotate them with the p-shift labels in Table 1; the `statistics` module provides functions for computing raw frequency counts, conditional probabilities, and the respective propensities; the `plotting` module provides a function, `frequency_treemap()`, that outputs a visual representation of the normalized frequencies for each participation shift as a treemap [17].

2.2. Input data to ParShift

To use *ParShift*, the user begins by importing the `parshift` library into a Python program and supplying an input file that contains turn-taking data from a group conversation. See Section 3 for an example. The input file must be formatted as a "comma-separated values" (CSV) file with the following columns:

`utterance_id:` unique utterance identifier.

`utterance:` utterance content.

`speaker_id:` unique speaker identifier.

`reply_to_id` | **`target_id`**: links the current utterance to a previous one by utterance ID or speaker ID, respectively.

2.3. Software functionalities

The `annotation` module converts the input CSV data file into a data frame where rows are validated conversation turns annotated with their corresponding p-shift. The following list describes the module's functions in detail:

`read_ccsv()` : this function reads the input CSV file, described in Section 2.2. The function forwards keyword arguments to Pandas's `read_csv()` function,[1] which it uses under the hood for converting and validating the CSV file into a Pandas data frame.

`conv2turns()` : aggregates utterance sequences in the input file corresponding to turn continuations where there is no change of target into a single turn, ensuring they are not considered as distinct participation shifts.

`annotate()` : calls `conv2turns()` to convert the conversation data frame returned by `read_ccsv()` into a Pandas data frame, where each row corresponds to a validated turn. Then, the function computes the p-shift label for each turn (e.g. AB-BA, $A0$-$X0$).

`pshift_class()` : returns the participation shift class, namely *Turn Claiming*, *Turn Continuing*, *Turn Receiving*, or *Turn Usurping* given a p-shift passed as parameter.

The second module, `statistics`, contains functions for extracting the frequency, probability, and conditional probabilities of each participation shift, namely:

`cond_probs()` : takes as input the Pandas data frame produced by `annotate()` and outputs another Pandas data frame containing the frequencies, probabilities and conditional probabilities for the p-shifts, which constitute the rows of the output data frame. This data frame is divided into two subgroups: those beginning with an undirected remark (A0-), and those beginning with a directed one (AB-). The column $P(S|D)$ (conditional probability of a p-shift, S, given a directed or undirected remark, D) contains a specific p-shift's frequency normalized by subgroup, and the $P(S|D,C)$ column (conditional probability of a p-shift, S, given a directed or undirected remark, D, and assuming there is a change of speaker, C) contains the same normalized frequencies in each subgroup, for each p-shift, but further conditioned on whether a change of speaker occurred.

`propensities()` : this function takes as input the Pandas data frame produced by `cond_probs()` to compute the three propensities proposed by Gibson [6], previously described in 1.2, outputting them in another Pandas data frame.

The `plotting` module has only one function,

`frequency_treemap` : takes as input the Pandas data frame generated by `cond_probs`, and returns a squarify treemap plot.

The `oo_parshift` module, not shown in Fig. 1, is an object-oriented application programming interface to *ParShift*, common in various Python libraries such as scikit-learn [18] or PyTorch [19]. The module provides the `Parshift` class, which contains the following methods:

[1] That is, values that, when passed into a function, are identifiable by specific parameter names.

Table 2
Example of an input CSV file from a synthetic conversation.

utterance_ids	speaker_id	utterance	target_id
1	1	Hey guys, have you ever considered that the earth might actually be flat?	None
2	2	What are you talking about? Of course the earth is round.	1
3	3	I've heard some people believe that the earth is flat. But there's no evidence to support that.	1
4	4	I think the idea of a flat earth is ridiculous. There's no way that could be possible.	1
5	5	The scientific evidence clearly shows that the earth is round.	1
6	6	I don't know where you're getting this idea from, but the earth is definitely not flat.	1
7	7	I'm pretty sure that's just a conspiracy theory. There's no reason to believe the earth is flat.	1
8	8	I think you're just trying to mess with us. Everyone knows the earth is round.	1
9	1	But have you ever actually seen the earth from space?	2
10	2	No, but plenty of astronauts have. And they've all said that the earth is round.	1
⋮	⋮	⋮	⋮
40	3	I think it's important to remember that not everyone has access to the same information and education. We need to be respectful of different perspectives.	None
41	4	That's a good point, Participant 3. It's important to approach these kinds of discussions with empathy and an open mind.	3
42	5	We should also be careful not to dismiss people's beliefs outright. It's better to engage with them and try to understand where they're coming from.	None
43	6	I agree with you Participant 5. It's important to listen to people and try to see things from their perspective.	5
44	7	But at the same time, we can't ignore scientific evidence and critical thinking. We need to find a balance.	None
45	8	Exactly. We should be open-minded, but also skeptical and informed.	None
⋮	⋮	⋮	⋮
95	2	Well said, Participant 1.	1
96	3	I think we've had a really productive conversation today. Thanks, everyone, for your contributions.	None
97	4	Agreed. It's been great to hear everyone's thoughts and ideas.	None
98	5	Yes, thank you all for being willing to engage in this discussion.	None
99	6	I'm looking forward to our next conversation.	None
100	7	Me too. It's always good to learn from each other and explore new ideas.	None

`process()` : takes the same input parameters as the `read_ccsv()` function, with an optional parameter N, which allows splitting the conversation into N parts. The function executes the entire annotation and analysis pipeline described in the previous modules and functions. Note: the optional parameter, N, should be greater than zero and less than five, its default value is $N = 1$.

`show_stats()` : outputs N Pandas data frames with the output of `cond_probs()`.

`show_plot()` : this method is an interface (in the current module) to the `frequency_treemap()` function. The number of plots in the figure will depend on the value N, passed to the `process()` method.

`get_propensities()` : this method is an interface (in the current module) to the `propensities()` function. If the value of N given in the `process()` is different from one, the returned data frame will have N rows. Otherwise, it will only have one row.

The three object-oriented method described above can receive an optional parameter, `filename`, which, if provided, will save the objects generated with that filename in the current directory.

3. Illustrative examples

The following example demonstrates how to use *ParShift* for the computational analysis of multi-party conversations. This example focuses on the artificial conversation in Table 2.

The first step is to install *ParShift* from the Python Package Index (PyPI) [20] — or directly from GitHub — and load it using the following instruction:

```
from parshift import Parshift
```

Next, a `Parshift` object is instantiated to represent the input conversation CSV file:

```
model = Parshift()
```

It is possible to compute the participation shifts by invoking the `process()` method, passing the path to the input CSV file. The second parameter, N, specifies that the input conversation should be divided in N equal parts (default $N = 1$). Researchers are often interested in exploring participation shifts in temporal segments, e.g., at the

Table 3
Output of the `annotate()` function, in the form of a Pandas data frame.

utterance_ids	speaker_id	utterance	target_id	pshift
[1]	1	Hey guys, have you ever considered that the earth might actually be flat?	None	
[2]	2	What are you talking about? Of course the earth is round.	1	A0-XA
[3]	3	I've heard some people believe that the earth is flat. But there's no evidence to support that.	1	AB-XB
[4]	4	I think the idea of a flat earth is ridiculous. There's no way that could be possible.	1	AB-XB
[5]	5	The scientific evidence clearly shows that the earth is round.	1	AB-XB
[6]	6	I don't know where you're getting this idea from, but the earth is definitely not flat.	1	AB-XB
[7]	7	I'm pretty sure that's just a conspiracy theory. There's no reason to believe the earth is flat.	1	AB-XB
[8]	8	I think you're just trying to mess with us. Everyone knows the earth is round.	1	AB-XB
[9]	1	But have you ever actually seen the earth from space?	2	AB-BY
[10]	2	No, but plenty of astronauts have. And they've all said that the earth is round.	1	AB-BA
⋮	⋮	⋮	⋮	⋮
[40]	3	I think it's important to remember that not everyone has access to the same information and education. We need to be respectful of different perspectives.	None	A0-X0
[41]	4	That's a good point, Participant 3. It's important to approach these kinds of discussions with empathy and an open mind.	3	A0-XA
[42]	5	We should also be careful not to dismiss people's beliefs outright. It's better to engage with them and try to understand where they're coming from.	None	AB-X0
[43]	6	I agree with you Participant 5. It's important to listen to people and try to see things from their perspective.	5	A0-XA
[44]	7	But at the same time, we can't ignore scientific evidence and critical thinking. We need to find a balance.	None	AB-X0
[45]	8	Exactly. We should be open-minded, but also skeptical and informed.	None	A0-X0
⋮	⋮	⋮	⋮	⋮
[95]	2	Well said, Participant 1.	1	A0-XA
[96]	3	I think we've had a really productive conversation today. Thanks, everyone, for your contributions.	None	AB-X0
[97]	4	Agreed. It's been great to hear everyone's thoughts and ideas.	None	A0-X0
[98]	5	Yes, thank you all for being willing to engage in this discussion.	None	A0-X0
[99]	6	I'm looking forward to our next conversation.	None	A0-X0
[100]	7	Me too. It's always good to learn from each other and explore new ideas.	None	A0-X0

beginning, middle or end of a conversation, see Section 1. In this example we use $N = 3$.

```
model.process("path_to_input_CSV_file", N=3)
```

3.1. Model components

The following snippet shows the model annotation, presented in Table 3, which is an intermediate step towards obtaining the conditional probabilities.

```
print(model.annotation)
```

To query the computations of participation shift frequencies, probabilities and conditional probabilities, the user can execute the following instruction:

```
model.show_stats(filename="output_file_name")
```

When the optional `filename` parameter is specified, the output is saved to one or more CSV files, depending on the value of N. Each file follows the structure of Tables 4, 5 and 6.

The `get_propensities()` method, used for obtaining the three propensities defined by Gibson [6], also accepts the `filename` parameter, as shown in the following snippet:

```
model.get_propensities(filename="propensities")
```

This function can also take a string parameter to specify the CSV file name to save the obtained result.

3.2. Graphical representation

Finally, a graphical representation of the participation shift frequencies can be generated as a treemap, as shown in Fig. 2. The following instruction saves the plot as a single 300 dpi PNG file:

```
model.show_plot(filename="output_file_name")
```

4. Impact

The *ParShift* library is built upon a framework designed for analysing oder and differentiation in multi-party conversations. The package provides a simple way to compute the relevant probabilities in Gibson's

Table 4
Output of the *ParShift* computations is represented as a Pandas data frame, specifically for the first third of the input example. The leftmost column contains the p-shift labels. The table is divided in two parts: the first part contains the p-shifts that correspond to undirected remarks (i.e. those starting with A0, known speaker to the group), while the second part contains the p-shifts for directed remarks. The four quantitative columns, from left to right, represent the raw counts for each p-shift, the overall empirical probability of observing a shift (S), the conditional probability of S given that the first turn was undirected (D = false) or directed (D = true), and the same probabilities further conditioned on whether there was a change of speaker (C). The Boolean values for C and D make up the remaining columns.

p-shift	Frequency	$P(S)$[a]	$P(S\|D)$[a]	$P(S\|D,C)$[a]	Change of Speaker (C)	Directed Remark (D)
A0-XA	2	0.06	0.29	0.29	True	False
A0-X0	4	0.12	0.57	0.57	True	False
A0-XY	1	0.03	0.14	0.14	True	False
A0-AY	0	0.0	0.0		False	False
AB-BA	10	0.31	0.4	0.4	True	True
AB-B0	0	0.0	0.0	0.0	True	True
AB-X0	2	0.06	0.08	0.08	True	True
AB-XA	0	0.0	0.0	0.0	True	True
AB-XB	7	0.22	0.28	0.28	True	True
AB-A0	0	0.0	0.0		False	True
AB-BY	6	0.19	0.24	0.24	True	True
AB-XY	0	0.0	0.0	0.0	True	True
AB-AY	0	0.0	0.0		False	True

[a] Rounded to two decimal places.

Table 5
Output of the *ParShift* computations is represented as a Pandas data frame, for the second third of the input example (recall $N = 3$ in this example).

Pshift	Frequency	$P(S)$[a]	$P(S\|D)$[a]	$P(S\|D,C)$[a]	Change of Speaker (C)	Directed Remark (D)
A0-XA	3	0.09	0.25	0.25	True	False
A0-X0	7	0.21	0.58	0.58	True	False
A0-XY	2	0.06	0.17	0.17	True	False
A0-AY	0	0.0	0.0		False	False
AB-BA	6	0.18	0.29	0.29	True	True
AB-B0	1	0.03	0.05	0.05	True	True
AB-X0	5	0.15	0.24	0.24	True	True
AB-XA	0	0.0	0.0	0.0	True	True
AB-XB	6	0.18	0.29	0.29	True	True
AB-A0	0	0.0	0.0		False	True
AB-BY	3	0.09	0.14	0.14	True	True
AB-XY	0	0.0	0.0	0.0	True	True
AB-AY	0	0.0	0.0		False	True

[a] Rounded to two decimal places.

Table 6
Output of the *ParShift* computations is represented as a Pandas data frame, for the third part of the input example (recall $N = 3$ in this example).

Pshift	Frequency	$P(S)$[a]	$P(S\|D)$[a]	$P(S\|D,C)$ [a]	Change of Speaker (C)	Directed Remark (D)
A0-XA	1	0.03	0.03	0.03	True	False
A0-X0	32	0.94	0.97	0.97	True	False
A0-XY	0	0.0	0.0	0.0	True	False
A0-AY	0	0.0	0.0		False	False
AB-BA	0	0.0	0.0	0.0	True	True
AB-B0	0	0.0	0.0	0.0	True	True
AB-X0	1	0.03	1.0	1.0	True	True
AB-XA	0	0.0	0.0	0.0	True	True
AB-XB	0	0.0	0.0	0.0	True	True
AB-A0	0	0.0	0.0		False	True
AB-BY	0	0.0	0.0	0.0	True	True
AB-XY	0	0.0	0.0	0.0	True	True
AB-AY	0	0.0	0.0		False	True

[a] Rounded to two decimal places.

Table 7
Output of the function that computes the propensities in Gibson's framework for each of the three segments (recall we set $N = 3$ in this example). The lower case n is used here to refer to each of the three segments of the input conversation.

	Turn-receiving	Targeting	Termination
$n = 1$	0.64	0.38	0.14
$n = 2$	0.48	0.31	0.17
$n = 3$	0.0	0.0	0.0

framework [6] systematically, therefore contributing to the study of order and differentiation in group conversations. Although Gibson created a model based on regular conversations in which people can address and reply to others, there is a gap in our understanding of some forms of computer mediated communication where addressing others only happens through explicit replies. Therefore, *ParShift* can be used not only for analysing face-to-face conversations based on Gibson's framework, but also for conversations mediated by e.g. online social platforms, mobile communication technologies.

To the authors' knowledge, *ParShift* is the first open-source package that implements the framework proposed by David Gibson [6]. The functions provided by this library are easy to understand, allowing researchers with no programming background to use it without major difficulties. Additionally, the provided functions implement specific steps of the conversational analysis process, meaning that a more experienced user can incorporate their own data and code at intermediate

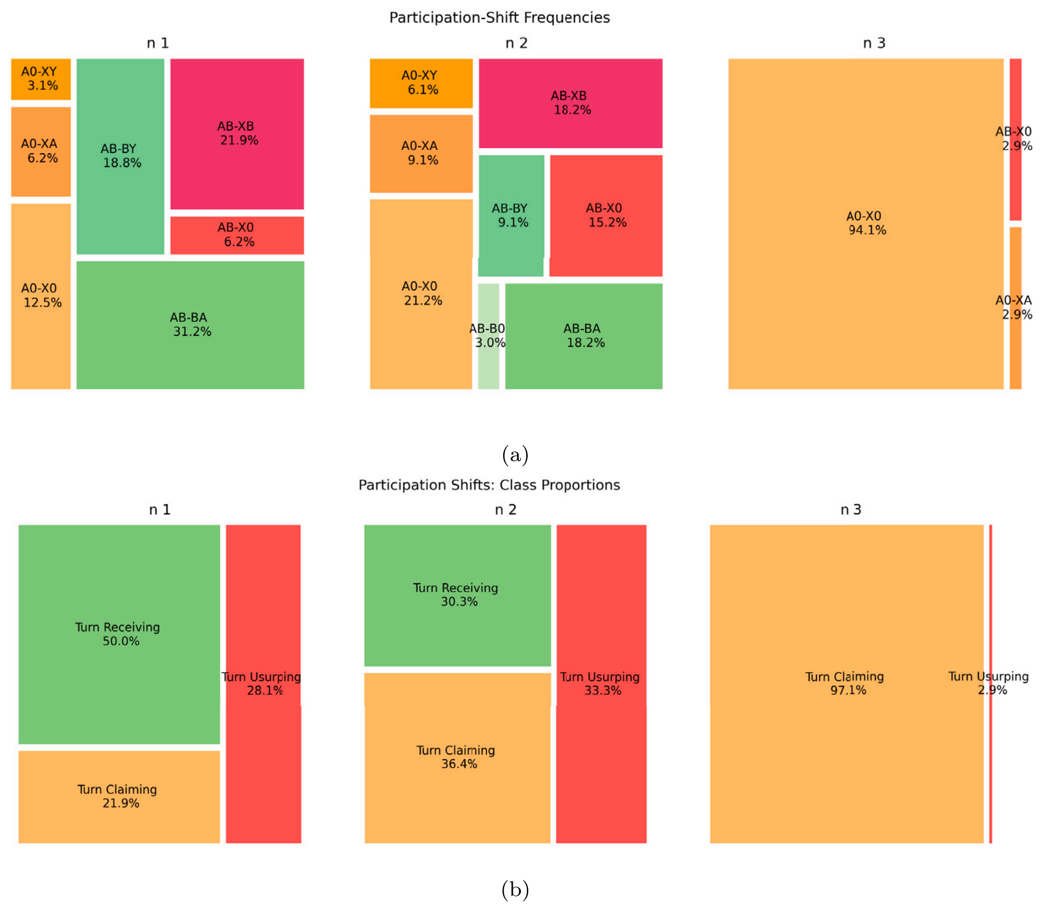

Fig. 2. *ParShift* graphical representation for (a) specific p-shift labels, and, (b) p-shift classes in the illustrative synthetic example described in the main text. The three plots in each section correspond to the first, second and last part of the analysed example in which we set $N = 3$.

steps of the *ParShift* pipeline. This way, it is not necessary to run the entire pipeline from the beginning, but rather start from a specific point in the sequence. Although based on Gibson's theory, this package also advances the study of a conversation by enabling the division of a group debate into N parts, as exemplified in Table 7, in order to explore the participation shifts at the beginning, middle or end of a conversation.

5. Conclusions

In this paper, we presented *ParShift*, a fully documented and unit-tested open-source Python library for studying emergent collective behaviours in multi-party conversations. Based on Gibson's Participation Shifts framework, the package offers procedural and object-oriented interfaces, accommodating users with varying degrees of technical knowledge. Given its uniqueness in this context, *ParShift* can be an important tool for the research community interested in small-group behaviour, conversational analysts and computational social scientists—particularly those interested in working with large datasets.

Declaration of competing interest

The authors declare no conflict of interest.

Data availability

No data was used for the research described in the article.

Acknowledgements

This work is supported by Fundação para a Ciência e a Tecnologia, FCT I. P. project “Factors for promoting dialogue and healthy behaviours in online school communities” with reference DSAIPA/DS/0102/2019 developed at the R&D Unit CICANT - Research Centre for Applied Communication, Culture and New Technologies at Universidade Lusófona, Portugal.

References

[1] Gu Jia-Chen, Tao Chongyang, Ling Zhen-Hua. Who says what to whom: A survey of multi-party conversations. In: Proceedings of the thirty-first international joint conference on artificial intelligence. 2022, p. 5486–93.
[2] Welles Brooke Foucault, González-Bailón Sandra. The oxford handbook of networked communication. USA: Oxford University Press; 2020.
[3] Sacks Harvey, Schegloff Emanuel, Jefferson Gail. A simplest systematics for the organization of turn-taking for conversation. Language 1974;50(4):696–735.
[4] Tuckman Bruce W. Developmental sequence in small groups. Psychol Bull 1965;63(6):384.
[5] Tuckman Bruce W, Jensen Mary Ann C. Stages of small-group development revisited. Group Organ Stud 1977;2(4):419–27.
[6] Gibson David R. Participation shifts: Order and differentiation in group conversation. Soc Forces 2003;81(4):1335–80.
[7] Cannon Bryan C, Robinson Dawn T, Smith-Lovin Lynn. How do we “do gender”? Permeation as over-talking and talking over. Socius 2019;5:2378023119849347.
[8] Olk Paul M, Gibbons Deborah E. Dynamics of friendship reciprocity among professional adults. J Appl Soc Psychol 2010;40(5):1146–71.
[9] Fuhse Jan A. Analyzing networks in communication: a mixed methods study of a political debate. Qual Quant 2023;57(2):1207–30.
[10] Engle Randi A, Langer-Osuna Jennifer M, McKinney de Royston Maxine. Toward a model of influence in persuasive discussions: Negotiating quality, authority, privilege, and access within a student-led argument. J Learn Sci 2014;23(2):245–68.

[11] Li Yiyang, Zhao Hai. EM pre-training for multi-party dialogue response generation. 2023, arXiv preprint arXiv:2305.12412.

[12] Hu Liru, Chen Gaowei. Exploring turn-taking patterns during dialogic collaborative problem solving. Instr Sci 2022;50(1):63–88.

[13] Gibson C Ben, Buchler Norbou, Hoffman Blaine, La Fleur Claire-Genevieve. Participation shifts explain degree distributions in a human communications network. PLoS One 2019;14(5):1–13.

[14] Kummerfeld Jonathan K, Gouravajhala Sai R, Peper Joseph, Athreya Vignesh, Gunasekara Chulaka, Ganhotra Jatin, et al. A large-scale corpus for conversation disentanglement. 2018, arXiv preprint arXiv:1810.11118.

[15] The pandas development team. pandas-dev/pandas: Pandas. Zenodo; 2023.

[16] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9(3):90–5.

[17] Shneiderman Ben, Wattenberg Martin. Ordered treemap layouts. In: IEEE symposium on information visualization 2001. INFOVIS 2001, IEEE; 2001, p. 73–8.

[18] Buitinck Lars, Louppe Gilles, Blondel Mathieu, Pedregosa Fabian, Mueller Andreas, Grisel Olivier, et al. API design for machine learning software: experiences from the scikit-learn project. In: ECML PKDD workshop: Languages for data mining and machine learning. 2013, p. 108–22.

[19] Paszke Adam, Gross Sam, Massa Francisco, Lerer Adam, Bradbury James, Chanan Gregory, et al. PyTorch: An imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems, vol. 32. Curran Associates, Inc.; 2019.

[20] Python Software Foundation. Python package index - PyPI. 2023, https://pypi.org/. [Accessed 27 May 2023].