The cpsystems LaTeX package

James Cooper, University of Auckland
jcoo092@aucklanduni.ac.nz

v0.12, 2019/12/31

Abstract

A package to assist authors writing about cP systems in typesetting their papers. It comprises a handful of environments and macros that are intended to ease writing about cP systems, and just as importantly, reduce the frequency of errors in their presentation. It is recommended to all authors using LaTeX to write about cP systems. Even if you don't want to use it, looking at the implementation details may give you some ideas for your own style.

1 Introduction

This package was originally created by James Cooper to help with typesetting a paper on cP systems (specifically one about modelling Belief Propagation in cP systems). The same commands had been copied from paper to paper, and across sections within papers, as most of them weren't made into proper LaTeX macros. This was, of course, extremely error prone, with formatting errors (and worse) sometimes making it into published articles.

The commands in this package in many cases are not necessarily less verbose than simply typing out the commands inside the macros. They are more 'robust'[1], however, in that by using the defined macros the exact same commands are applied each time so there is greater consistency. If the macro is mistyped, the (La)TeX engine itself will report the error. They also hopefully should add greater structure to a paper and prove useful in editing.

2 Usage

Put text here. Note that this section is currently woefully under-complete.

`cpruleset` Goes inside a `cprulesetfloat`

`cpobjects`

`\cprule` `\cprule {`⟨*Starting state*⟩`} {`⟨*Input objects*⟩`} {`⟨*Mode of operation*⟩`} {`⟨*Ending state*⟩`} {`⟨*Output objects*⟩`}`

`\cppromoter`

[1] Note that LaTeX has its own, different, concept of "robust".

\cpinhibitor
\cpsend
\cprecv
\cpfunc
\cpobjectsline
\cpterm

3 Implementation

This section presents the actual implementation of the package. For the most part you probaly won't need to refer to it, but every so often you might, especially to work out some error that LaTeX is throwing at you, based on what the commands defined within become once they have been substituted into your document.

```
%

\RequirePackage{array}
\RequirePackage{framed}
\RequirePackage{changepage}
\RequirePackage{amsmath}
\RequirePackage{trimspaces}
\RequirePackage{newfloat}

\newcounter{cpsystems@RuleNum}
```

cprulesetfloat A floating environment inside which cpruleset environments are to be placed. This 'wrapping' float provides both the floating capability, as well the ability to caption, label and reference cprulesets.

```
\DeclareFloatingEnvironment[name=Ruleset,within=none]{cprulesetfloat}
```

cpobjectsfloat A floating environment inside which cpobjects environments are to be placed. This 'wrapping' float provides both the floating capability, as well the ability to caption, label and reference cpobjects.

```
\DeclareFloatingEnvironment[name=Objects Group,within=none]{cpobjectsfloat}
```

cpruleset A wrapper environment in which cprules are listed, and which mimics the usual style of presentation for rules: A lined box with the rules inside it.

```
\newenvironment{cpruleset}
{\begin{framed}\begin{adjustwidth}{-1.0em}{-1.0em}
\renewcommand{\arraystretch}{1.0}\[\begin{array}{lllllr}}
{\end{array}\]\end{adjustwidth}\end{framed}}
```

cpobjects A wrapper environment in which cpobjectlines are listed, imitating a style used in the past: A lined box with lines of cP systems objects defined inside it. Primarily used for illustrating examples.

```
\newenvironment{cpobjects}{\begin{framed}}{\end{framed}}
```

\cprule For writing out a rule inside a cpruleset environment. Required arguments are, in order, beginning state name; LHS of rule; the label to be applied to the arrow; the ending state name; the RHS of the rule.

```
\newcommand{\cprule}[5]{
    \refstepcounter{cpsystems@RuleNum}
    \trim@spaces@noexp{#1 & #2 & \rightarrow_{#3} & #4 & #5
     & (\arabic{cpsystems@RuleNum})\\}
}
```

\cppromoter For specifying promoters as part of a rule.

```
\newcommand{\cppromoter}[1]{
\trim@spaces@noexp{& & & & ~ \hspace{0.5cm} ~ | ~ #1 & \\}
}
```

\cpinhibitor

```
% For specifying inhibitors as part of a rule.
\newcommand{\cpinhibitor}[1]{
\trim@spaces@noexp{& & & & ~ \hspace{0.5cm} ~ \neg ~ #1 & \\}
}
```

\cpsend Encapsulate a 'send' in cP systems. First argument is the object(s) to be sent, and the second argument is the name of the channel the object(s) shall be sent on.

```
\newcommand{\cpsend}[2]{
\trim@spaces@noexp{\{#1\}!_{#2}}
}
```

\cprecv Encapsulate a 'receive' in cP systems. First argument is the object(s) to be received, and the second argument is the name of the channel the object(s) shall be received on.

```
\newcommand{\cprecv}[2]{
\trim@spaces@noexp{\{#1\}?_{#2}}
}
```

\cpfunc Command for declaring a cP systems functor. The first argument is the symbol for the functor itself, and the second argument is the objects contained inside the functor.

```
\newcommand{\cpfunc}[2]{
\trim@spaces@noexp{#1\big(#2\big)}
}
```

\cpobjectsline Used for presenting a group of objects, inside a cpobjects environment.

```
\newcommand{\cpobjectsline}[1]{
\[#1\]
}
```

\cpterm Explanation TBF

```
\newcommand\cpterm[2]{%
\item[$#1$]#2.}
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Change History

0.11
  General: Converted to DTX file . . 1

0.12
  General: Completed documentation for first DTX format attempt 1