IPSJ全国大会用クラスファイルの使い方

山田 太郎 † 山田 花子 †† 鈴木 太郎 ††,†††

† 名古屋大学工学部電気電子・情報工学科 †† 名古屋大学大学院情報学研究科
††† 名古屋大学大学院情報科学研究科

1 はじめに

本稿では，情報処理学会全国大会に投稿する論文用のクラスファイル “ipsj.cls” の使用法を説明する．なお，本稿を作成するためのソースファイルは本クラスファイルのテンプレートも兼ねている．詳しくは後述するが，コメントで “Describe ...” と書いてある箇所を変更することで，ソースファイルをそのまま論文作成に使用できる．

2 使い方

基本的な書き方は LaTeX 2ε で文章を書く際と同じであり，タイトル・著者・所属の指定方法のみ拡張している．

タイトルを指定する際は，以下のように日本語タイトルを第 1 引数に，英語タイトルを第 2 引数に指定する．

```
\title{日本語タイトル}{english title}
```

著者を入力する際は，第 1 引数に日本語名を，第 2 引数に英語名を，第 3 引数に後述する所属ラベルを指定し，人数分列挙する．

```
\author{日本語名}{english name}{L}
```

所属が複数ある場合は，以下のように半角カンマ（,）区切りでラベルを列挙する．

```
\author{日本語名}{english name}{L1,L2}
```

所属を入力する際は，第 1 引数にラベルを，第 2 引数に日本語での所属を，第 3 引数に英語での所属名を指定する．

```
\affiliation{L}{日本語所属}{english affl.}
```

なお，“\author” で使用したラベルが “\affiliation” に存在しない場合，エラーが発生するため注意すること．

また，著者と所属の表示順は，入力された順番にそのまま表示される．これは参照符についても同様であり，著者入力時のラベルで指定した順番に付与される．したがって，所属の出力順と参照符の付与順に関しては手動であわせる必要がある（自動化できるとは思いますが，面倒なのでたぶんやりません）．

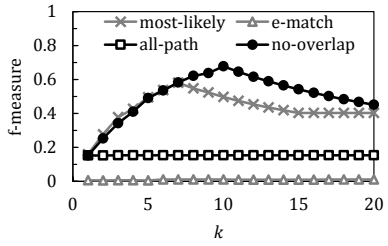

図 1 図の例

3 出力例

本章では，章節項や図表，アルゴリズムの出力例を示す．なお，本クラスファイルは “jsarticle.cls” の拡張であり，各レイアウトは変更していないため，“jsarticle.cls” を用いた際と同様の出力結果が得られる（はずである）．したがって，章節項のレイアウトを変更したい場合は，“jsarticle.cls” での変更例を参考にすればよい．

3.1 節

節の出力例．

3.1.1 項

項の出力例．

■見出し付きパラグラフ パラグラフの出力例．

3.2 図の出力例

図の出力例を図 1 に示す．

以下，図を使用する際の注意点を列挙する．

■PDF ファイルを使う： 以前は論文を DVI ファイルで提出するのが主流だったため図も EPS ファイルで作成していたが，現代ではほぼ間違いなく PDF ファイルの提出を求められる．したがって，PowerPoint などで作成した図を直接 PDF ファイルとして保存し，挿入すればよい．ただし，PDF ファイルを挿入する際は，画像のバウンディングボックスについての情報を持つ XBB ファイルを用意する必要がある．無い場合はコンパイル時に自動で用意されるが，数が多くなると時間がかかるため，あらかじめ “extractbb” コマンドで作成した方がよい．また，PowerPoint で PDF ファイルを作成した場合，ファイルの容量が無駄に大きくなる．そのため，GhostScript を用いて容量を削減することを勧める．なお，上記 2 点に関しては処理を簡便にするために Bash スクリプトを作成したので，活用してほしい．Bash が動く環境（MINGW など）があれば，以下のコマンドで指定したディレクトリ内にある全 PDF ファイルの容量削減と XBB ファイルの作成が実行できる．

```
./prepare-pdf.sh <figures_dir>
```

How to use the class file for the national convention of IPSJ

Tarou Yamada†, Hanako Yamada††, and Tarou Suzuki††,†††

†Department of Information Engineering, School of Engineering, Nagoya University

††Graduate School of Informatics, Nagoya University

†††Graduate School of Information Science, Nagoya University

表 1　表の例（“booktabs.sty” 使用）

(1, 1)	(1, 2-3)	
(2-3, 1)	(2, 2)	(2, 3)
	(3, 2)	(3, 3)

Input: $I = \{1, 2, 3, 5, 7, 11\}$　// 特に意味のない素数
Output: *sum*
1 $sum = 0$
2 **foreach** $i \in I$ **do**　// 特に意味のない for ループ
3 　$sum = sum +$ `fibonacci(`i`)`
4 **Function** `fibonacci(`n`)`　// フィボナッチ数列
5 　**if** $n = 0$ **then**
6 　　**return** 0
7 　**else if** $n = 1$ **then**
8 　　**return** 1
9 　**else**
10 　　**return** `fibonacci(`$n - 2$`) + fibonacci(`$n - 1$`)`

図 2　アルゴリズムの例

■グレースケールで作る：　PDF ファイルでの提出が主流となったため，基本的に図で色を用いても問題はない．しかし，印刷時に読み手が必ずしもカラー印刷を行うとは限らないのに加え，人によっては色盲により書き手が色に込めた意図を汲み取れない可能性もある．そのため，図は始めからグレースケールで作成するか，グレースケールで印刷されても十分に伝わるように作るべきである．

3.3　表の出力例

表 1 に表の出力例を示す．

表に関しては，“booktabs.sty” の使用を勧める．基本的に，表は罫線の少ない方が見やすい．しかし，LaTeX 2ε 標準の表では，列の境目を縦の罫線で表すしかない．そこで，“booktabs.sty” を使用することで，列の境目を横罫線の切れ目として表せる．また，表の上下で使う罫線を太く，中で使う罫線を細くすることで，よりすっきりとした表が作れる．詳しい使い方については，複数列・複数行に渡るセルの書き方とあわせて表 1 で使用しているため，ソースファイルを参照してほしい．

3.4　アルゴリズムの例

アルゴリズムの例を図 2 に示す．

アルゴリズムに関しては，“algorithm2e.sty” の使用を（個人的に）勧める．調べると “algorithmicx.sty” などの方がよく情報が出てくるが，“algorithm2e.sty” の方がすっきりとして見やすいアルゴリズムを書けると感じる．記法にはやや癖があるものの，基本的な書き方は図 2 のソースを見ればわかると思うため，アルゴリズムを書く必要がある際は候補に考えてほしい．また，公式のドキュメントもしっかりと書かれているため，分からない点があれば参照するとよい．

3.5　参考文献の例

参考文献を参照した際の例 [1–5]．“cite.sty” を使用しているため，連続して参照すると連番として表示される．

LaTeX 2ε で参考文献を列挙する際は BibTeX の使用を勧める．あらかじめ参考文献を列挙した “.bib” 拡張子のファイルを用意し LaTeX 2ε ソース内でインポートすることで，コンパイル時に自動的に参照の解決が行われる．

4　おわりに

本稿では情報処理学会全国大会用のクラスファイル “ipsj.cls” について解説した．また，クラスファイルの基本的な使用方法に加えて，図表やアルゴリズムの書き方，挿入の仕方についても簡単に述べた．

なお，本稿で述べた以外にも，LaTeX 2ε で論文を書く際のコツはいくつかある．ただし，慣れてくると無意識で行っているものも多いため，全てを説明するのは難しい．そこで，本稿を作成する際に使用したソースファイルでは，著者が普段論文を書く際に使用しているスタイルファイルやマクロをそのまま列挙した．本文中での使われ方とあわせて，よければ参照してみてほしい．

参考文献

[1] A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques, and Tools*. Addison Wesley, 2006.
[2] Y. Diao, P. Fischer, M. J. Franklin, and R. To, “YFilter: Efficient and scalable filtering of XML documents,” in *Proc. 18th ICDE*, pp. 341–342, 2002.
[3] R. Kowalski and M. Sergot, “A logic-based calculus of events,” *New Generation Computing*, vol. 4, no. 1, pp. 67–95, 1986.
[4] J. E. Hopcroft, “An $n \log n$ algorithm for minimizing states in a finite automaton,” tech. rep., Stanford University, 1971.
[5] Grep - GNU Project - Free Software Foundation: `https://www.gnu.org/software/grep/` (accessed: December 17, 2017).