**Point-to-point list**

We have carefully read all reviewer comments and addressed each single one appropriately. Most reviewer comments are not related the manuscript, but to the application itself. Thus, only small changes have been made to the manuscript, but great changes have been made to the application and related documents. The main changes are:

* The application has been bugfixed, improved and released in version 1.1 now.
* We have enhanced the documentation (directly in the application, as well as on the official homepage)
* A Javadoc KEGGtranslator API with examples for using it, is now available on the official homepage
* We created a list of “Frequently Asked Questions” (FAQ), which we put on the homepage
* The manuscript now has a better matching title and includes an additional reference to a new program that, like others, also does conversion from KEGG to SBML only. We included this reference to have a more complete list of related works.

In the following, we would like to discuss all reviewer comments in detail. Therefore, we colored the reviewer comments in red and our answers/actions in blue below.

Reviewer: 1

Comments to the Author

The authors present a simple but very useful software tool that can convert the KEGG pathway information from its original format to several XML and image formats.

The program has a user friendly graphical interface and can be launched from a browser via the Java Webstart. The program does not implement particularly sophisticated methods but I consider it a very useful application.

Thank you.

Reviewer: 2

Comments to the Author

1) General comments:

I was excited to read the abstract, but I'm missing quite some other formats: A true translator would incorporate additional formats such as BioPax or Ondex.

We are focusing on mainly two kinds of translations: a graphical translation, for generating graph structures (like the GML or GraphML format) and a functional translation that allows for easy simulations of KEGG pathways (SBML format). We have chosen GML, GraphML and SBML as target formats, among the great range of data formats, because these are very popular. In this way, users can optionally consider a great variety of further translation applications to translate KGML files first to SBML or GML and then to almost any format. Thus, we indirectly support BioPax and Onex:

BioPax can be generated by translating to SBML and using the official EBI “SBML to BioPax” translator, available at <http://www.ebi.ac.uk/compneur-srv/sbml/converters/SBMLtoBioPax.html> or PaxTools, available at <http://www.biopax.org/>.

Ondex also supports SBML files directly in their official application (see <http://www.ondex.org/formats.html>). Thus, KEGG2Ondex can be performed fast and efficiently by using KEGGtranslator to translate to SBML and importing the document into the Ondex application.

Since this is a Java application and claims to produce structural pathway information, it would also be nice to read something about how it could integrate with existing Java graph libraries such as JUNG.

We thank the reviewers for mentioning this graph library. KEGGtranslator includes yFiles as internal data structure for graphs. JUNG would be another option, which could be used for this task. There are many graph libraries available, all of which have classes for “Nodes”, “Edges” and several visualization/annotation classes. Thus, transitions from one data structure to another can easily be performed by iterating over nodes from one data structure and creating nodes in the other one. Having yFiles as internal graph library is completely sufficient, because there is no need to internally use multiple graph drawing libraries.

Furthermore, as mentioned above, we have chosen the most popular output formats. JUNG provides features to import GraphML files (see <http://jung.sourceforge.net/doc/api/edu/uci/ics/jung/samples/GraphFromGraphMLDemo.html>). Thus, KGML files can be used in JUNG, by generating a GraphML file with KEGGtranslator and importing it into JUNG.

Major:

While the paper mentions command-line options, I don't see this documented any further on the website or in the manuscript.

All command-line options are documented directly in the application and can be accessed by

* pressing “F1”
* selecting “Help” -> ”Online Help”
* starting the application with “-help” or “-?”

Moreover, all options are described in the documentation section of the KEGGtranslator homepage (see <http://www.cogsys.cs.uni-tuebingen.de/software/KEGGtranslator/doc/index.htm>).

Yet the automated processing of a number of KGML-files would be very useful for users. True, a GUI allows for exploratory data examination; but once a protocol has been determined for a specific pathway, usually one wants to repeat the protocol for several others. At that point, the GUI becomes rather cumbersome to operate.

That’s completely true. KEGGtranslator supports batch processing of multiple files and complete directories with subdirectories. Users can even specify every translation option available in the GUI, for the batch translation. Simply call KEGGtranslator with --input <directory> to batch convert all files in that directory. We changed the documentation of KEGGtranslator to also mention this detail.

Although such a direct batch mode seems more convenient for the user, there is generally the possibility to write a loop on the command line which runs over multiple files in a certain directory and calls KEGGtranslator for each file individually. This approach would be more consistent with the UNIX philosophy that one program processes one file but loops and pipes may connect these.

If you used it, you may want to mention something in the manuscript about MVD software engineering techniques that you used.

Unfortunately, we don’t know what is meant with MVD and also Google does not bring up valuable results in connection with software development. If you meant MVC (model view controller) with that, we have used this pattern, but we want to focus the paper rather on the actual translation of KEGG files than on software engineering techniques (the paper is restricted to two pages only).

And if you do have a separate library that can be brought into an IDE such as Eclipse, you should place its JavaDoc online on your website, along with code examples of how to use it.

Excellent idea! We generated a Javadoc and various code examples of how to use it at <http://www.cogsys.cs.uni-tuebingen.de/software/KEGGtranslator/doc/Javadoc/>. This is now also mentioned in the FAQ and the documentation.

The program at current doesn't seem to incorporate sufficient additional value compared to existing tools. Two examples: [1] KEGG2SBML is a simple Perl-script provided through the sbml.org website; Operating KEGGTranslator is slower to boot and more complex to operate in contrast. [2] It seems that any translator needs to incorporate and combine most of the already existing individual tools. Most noteworthy: R seems to be left out, while it already exists elsewhere. So this tool is NOT the all-emcompassing long sought-after KGML-solution that the authors claim it is.

All previously published translators only translate KGML files to one output format (mostly SBML) and purely use the content, available in the XML document. KEGGtranslator supports multiple file formats and provides some essential functionality that has not been included in any other translator before:

* KEGG’s XML documents often contain incomplete reactions that miss some reaction participants (see Figure 1 of the publication). We implemented an autocorrection feature that automatically checks each reaction and adds missing reactants to reactions. This is extremely helpful, especially if the reactions of the pathways should be used in further simulations.
* Original node names are barely readable. The following example is taken from the “MAPK signaling pathway” for homo sapiens (KEGG ID “path:hsa04010”):

|  |
| --- |
| <entry id="10" name="cpd:C00076" type="compound"  link="http://www.kegg.jp/dbget-bin/www\_bget?C00076">  <graphics name="C00076" fgcolor="#000000"  type="circle" x="314" y="175"/>  </entry> |

This compound, named “C00076”, represents “Ca2+”. KEGGtranslator automatically queries the KEGG database and employs a readable name (“Ca2+” in this example) for nodes. Other KEGG converters solely work on the content, provided in the XML document.

* Annotations of all pathway components with unambiguous, standardized MIRIAM URNs have been added. These functionalities have not been published before. Hence, KEGGtranslator adds many identifiers to each element in a pathway. This includes, e.g.,

identifiers for several KEGG databases, EntrzGene, Omim, HGNC, Ensembl, UniProt, Chebi, Drugbank, Gene Ontology, PubChem, 3Dmet, NCBI Taxonomy, PDBeChem, GlycomeDB, LipidBank, and many more.

* Despite the machine-readable standardized identifiers, KEGGtranslator adds human readable comments to each node. This includes for each element (gene, pathway, relations, reactions, etc.):
  + official descriptions
  + links to the original entry, describing the element on the KEGG homepage
  + All Synonyms (e.g., for genes)
  + Mass and Chemical formula for applicable elements (e.g., small molecules)
  + … and many more
* In SBML documents, there exist SBO terms to further describe the type of a component. A best matching SBO term is automatically added for each element by KEGGtranslator.
* In a graphical translation, the KEGGtranslator GUI allows graph manipulations like moving nodes and edges or changing the size of nodes, etc.
* … and some other options, that are also mentioned and described in the publication (e.g., removing white gene nodes, which represent biological entities that are unrelated to the current organism)

We briefly mention some of those features and a comparison to other translators (like KEGG2SBML, KEGGconverter and KEGGgraph) at the end of Section 1 in our publication. All these key features of KEGGtranslator are mentioned in Section 2 of the publication.

Besides this, we are publishing a Java application, which has the great advantage of being nearly platform independent. We understand that there are researchers, working with Perl, R, Matlab and many other environments, but we simply cannot provide this application for all different programming languages. Nevertheless, for these purposes, R, Perl, Matlab, etc. implement runtime capabilities that allow the execution of external applications. And for environments that don’t implement runtime capabilities, simple shell scripts can still be employed to, e.g., first, translate files and second, load the new file into the environment. Thus, one can execute KEGGtranslator directly from these applications and further use the resulting, converted pathway file in the respective environment. We would like to refer to the general problem of the existence of different programming environments here and their solution.

It is impossible to chase all existing bioinformatics data formats It may be more useful to re-design this application as a plugin for existing platforms, such as Cytoscape or CellDesigner (which would allow even more interaction than the current version already supports).

As mentioned above, the application can be used much more generic, if it can directly be executed from the command-line or via the GUI. If we would release it as a plugin for Cytoscape or CellDesigner, only researchers using Cytoscape or CellDesigner would be able to use it. Whereas a stand-alone application can be used by everybody. KEGGtranslator offers conversion to GML files that can seamlessly be opened by Cytoscape. And conversion to SBML files with CellDesigner annotations that can be opened with CellDesigner is also supported directly by KEGGtranslator. Thus, we are supporting these two (and by not developing a plugin even more) applications.

Minor:

When starting the application through Java Web start, a warning pops up: "The application's digital signature cannot be verified. Do you want to run the application?". You may want to obtain a security certificate from a Verisign in order to resolve this.

Unfortunately, these certificates are not cheap and our Institution does not provide a certificate.

When you mention applications for your tool, they should be more defined and referenced with some examples from other groups (e.g. when you talk about stoichiometry). Right now it all seems very vague.

That’s completely true. Unfortunately, a Bioinformatics applications note is limited to two pages and we decided to focus rather on the program itself, than on possible applications. Thus, we mentioned them briefly but not explain them in more detail.

Some examples for stoichiometric analyses:

* A stability analysis: This is a steady-state analysis based on the product of the stoichiometric matrix and the flux vector. To apply this method, kinetic equations need to be defined for the system first, which can be derived automatically by loading the created SBML file into SBMLsqueezer (Dräger et al. 2008) and estimating the model parameters for the steady-state. For details on stability analysis, the book of Heinrich and Schuster (1996) is a very good reference.
* A flux balance analysis: This analysis is based on the stoichiometric matrix only. No kinetic information is required. Several tools, e.g., FASIMU (Hoppe et al. 2011) are available to conduct such analyses.
* An automatic check if all reactions are balanced (we noticed that in KEGG particularly protons are missing in some reactions). This can be implemented easily in Java or other languages.

Dräger et al. 2008: <http://www.biomedcentral.com/1752-0509/2/39>

Hoppe et al. 2011: <http://www.biomedcentral.com/1471-2105/12/28>

Heinrich and Schuster 1996: <http://www.amazon.de/Regulation-Cellular-Systems-Reinhart-Heinrich/dp/0412032619/ref=sr_1_1?ie=UTF8&s=books-intl-de&qid=1305636738&sr=8-1>

At least the creation of kinetic equations using the tool SBMLsqueezer (Dräger et al. 2008) based on the rich annotations provided by KEGGtranslator is cited on page two of the paper. Due to the strict space limitation it is not possible to cite this paper more than once. People interested in stoichiometric analysis will find relevant publications on this by querying PubMed on these keywords.

Reviewer: 3

Comments to the Author

The authors describe a java application for visualising and translating the KEGG pathway database kgml files. The application is interesting: it has a gui, supports many output formats and allows to edit the kegg maps. The web site dedicated to the application is nice and a (brief) documentation is provided.

I tried the tool with some pathways (e.g rno04630 rattus norvegicus jak-STAT signalling pathway) and I noticed that while the "graphical translation" works fine, the "functional translation" often produces an sbml file that DOES NOT contain any reactions, but it contains only the list of compartments and the list of species: this is an important loss of information that causes the SBML file to be not so useful.

Major

The authors should improve the "functional translation" in order to obtain sbml files with a list of reactions. How is it possible to use an sbml file for dynamic simulations without any reactions? Alternatively, the authors should explain better the issues that make this operation not possible: what are the limitations of the tool? when is it possible to obtain a "complete" sbml file for simulation?

----------------------------------------

Thank you for this interesting feedback. KEGG provides pathways as KGML formatted XML-documents. KGML files contain three main elements:

* entries (which are genes, compounds, references to other pathways, etc.)
* relations (which are “arrows” in a pathway)
* reactions (which are functionally important).

In graphical translations (e.g., KGML2GML, KGML2GraphML), all relations are translated to corresponding arrows. This results in a graph-representation of the pathway that can be visualized and used for further graph operations or as pathway picture.

In functional translations (e.g., KGML2SBML), all reactions are translated to SBML reactions. This results in a model that can, e.g., later be annotated with kinetics and simulated.

Entries are always translated and annotated to nodes (graphical translation) or species (functional translation). Unfortunately, KEGG does not provide reactions for all pathways. For example, the mentioned pathway “rno04630 rattus norvegicus jak-STAT signalling pathway” does only contain relations and entries. The SBML specifications only describe quantitative interactions, no qualitative relationships like these. This means, in SBML it is not possible to state “A interacts with B”. We can only express “A reacts to B”. However, “A interacts with B” does not specify a reaction in which both elements are involved. Creating some reaction such as “A + B → ø” would not imply the same meaning. A dynamic simulation of these processes would then imply that A and B are removed from the system, which is not intended by the original statement. Currently, an extension package for SBML Level 3 is under development to also incorporate qualitative relationships into the SBML standard. However, this extension package is not yet completed and no software is hence available supporting it. One way to perform dynamic simulations on networks that contain qualitative relationships would be to mathematically describe these relationships in terms of rules in SBML (algebraic rules, assignment rules, or rate rules). However, this has to be done by the user because these mathematical expressions require the insertion of specialized mathematical expressions that cannot be derived automatically and particularly not in a general way in a tool such as KEGGtranslator.

Nevertheless, the “Phosphatidylinositol signaling system” pathway for rat (available at <ftp://ftp.genome.jp/pub/kegg/xml/kgml/non-metabolic/organisms/rno/rno04070.xml>) is an example for a rat pathway that also contains reactions.

We are thankful for pointing out this issue that this is really not obvious to first time users. Thus, we changed the documentation to more clearly reflect this detail and added this to our list of frequently asked questions. The publication already mentions this detail briefly: “[…] the focus in functional translation lies on the reactions in KGML documents, whereas graphical representations concentrate on relations between pathway elements”.