HHS Public Access
Author manuscript
SoftwareX. Author manuscript; available in PMC 2025 May 23.

Published in final edited form as:
SoftwareX. 2024 December ; 28: . doi:10.1016/j.softx.2024.101913.

MPAT: Modular Petri Net Assembly Toolkit

Stefano Chiaradonna[a], **Petar Jevtić**[a,*], **Beckett Sterner**[b]

[a]School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ, United States of America

[b]School of Life Sciences, Arizona State University, Tempe, AZ, United States of America

Abstract

We present a Python package called Modular Petri Net Assembly Toolkit (MPAT) that empowers users to easily create large-scale, modular Petri Nets for various spatial configurations, including extensive spatial grids or those derived from shapefiles, augmented with heterogeneous information layers. Petri Nets are powerful discrete event system modeling tools in computational biology and engineering. However, their utility for automated construction of large-scale spatial models has been limited by gaps in existing modeling software packages. MPAT addresses this gap by supporting the development of modular Petri Net models with flexible spatial geometries.

Keywords

Petri Nets; Discrete event simulation; Spatial modeling; Spike; Snoopy

---

Code metadata

Current code version	0.0.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-24-00376
Permanent link to Reproducible Capsule	https://github.com/schiarad2354/Petri_Net
Legal Code License	Non-commercial license: CC BY-NC 4.0
Code versioning system used	git
Software code languages, tools, and services used	Python, Spike
Compilation requirements, operating environments & dependencies	Python 3.8, GeoPandas, Pandas, NumPy
If available Link to developer documentation/manual	https://github.com/schiarad2354/Petri_Net

---

*Corresponding author. pjevtic@asu.edu (Petar Jevtić).

CRediT authorship contribution statement
Stefano Chiaradonna: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation. **Petar Jevtić:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization. **Beckett Sterner:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Support email for questions	pjevtic@asu.edu

1. Motivation and significance

1.1. Introduction

Many physical and biological systems can be understood as processes involving varying types of interactions among spatially distributed modular components. Mathematical formalisms for discrete event systems (DES) have been widely used to represent the dynamics of complex systems and resilience modeling [1]. Petri Net models are a prominent approach to DES simulation that is especially suitable for modeling and analyzing systems characterized by concurrent and distributed processes using directed graphs [2–4]. Several software packages currently support Petri Net simulation modeling, including Spike [5] and GPenSim [6]. However, to our knowledge, existing software does not provide functionalities for the automated construction of large-scale, modular Petri Net systems with flexible spatial geometries.

1.2. Heterogeneous information layers

Automated construction of large-scale Petri Net models is an important stepping stone for building complex systems compromised of multiple heterogeneous models over space and time dependent on various information layers. For instance, when it comes to information layers, modelers in ecology leverage granular spatio-temporal terrain data for modeling microclimates [7] or leverage heterogeneous habitat spatial data for modeling animal geographic patterns [8]. Similarly, for modeling natural hazards, meteorological data, for example, is crucial for rapid urban risk mapping from floods [9], earthquakes [10], tornados [11], and hurricanes [12]. In addition, insurers also utilize meteorological data for predicting hail claims [13], improving agriculture risk assessment [14], developing weather insurance for specific crops [15], etc. In other instances, modelers may leverage multiple, heterogeneous information layers simultaneously. For example, Boyle et al. [16] leveraged historical wind speed, social vulnerability, and power flow for power loss mitigation. Similarly, Chandrappa et al. [17] utilized humidity, precipitation, soil moisture, solar radiation, wind direction, temperature, etc. To integrate the wide range of information layers that modelers may utilize, we introduce the Modular Petri Net Assembly Toolkit (MPAT) to automate the construction of modular, spatial Petri Net models to be executed using the Spike software.

1.3. Overview of Petri Nets

Precisely, a Petri Net can be defined as a tuple (P, T, F, M_0), where P is a set of places, T is a set of transitions, and P and T are disjoint sets. The function $F: (T \times P) \cup (P \times T) \to N$ assigns a weight to each arc in the Petri Net. In particular, M_0 denotes the initial condition of the Petri Net as a function from the set of places to the non-negative integers. The arcs connect a place to a transition or a transition to a place. The input places are the places connected by an arc going to a transition, and the output places are the places connected by an arc coming from a transition. Arcs can be assigned weights with non-negative integer values. A transition becomes enabled when the number of tokens in the input place is at least the arc's weight. When the transition fires, it will consume a total number of tokens matching the arc weight from each input place and produce a total number of tokens matching the weight of each arc connecting to an output place. The distribution of tokens over the place represents a configuration or marking of the net.

Petri Nets continue to be a popular modeling choice for a wide range of researchers in varying domains. As a testament to the widespread adoption of Petri Nets, a Google scholar search with the exact phrase "Petri Net" showed 17,100 results since 2020 (accessed on August 26, 2024). In fields where spatial and temporal considerations are crucial, Petri Nets have proven particularly effective. For instance, a Google Scholar search using "Petri Net" AND "Spatial" keywords showed 3480 articles.[1] Additionally, the search results yielded 2420 results for the keywords "Petri Net" AND "Resource Allocation", 714 results for "Petri Net" AND "Biochemical", 635 results for "Petri Net" AND "Ecology", and 91 results for "Petri Net" AND "Wildfire". Recent studies have also showcased their utility in epidemiology [18–20], resilience engineering [21], and computer engineering, particularly in spatial networking protocols for parallel and distributed systems [22, 23]. Despite this versatility of use of Petri Nets, to our knowledge, there is no assembly tool that can take different information layers relevant to an individual Petri Net and give their internal interconnectivity assessment of large-scale Petri Net. Typically, the only choice available to modelers is to use Colored Petri Nets.

Colored Petri Nets are popular for modeling spatial systems with a high level of symmetric structure because they simplify the model representation in terms of stacks of repeated components distinguished by different color labels [24–27]. However, Colored Petri Nets also have limitations. They require considerable expertise in accurately modeling them, which can be a barrier for new users or those without a strong background in Petri Net methodology. Moreover, the benefits of using colored Petri Nets fall short when representing large models with heterogeneous structures and parameter settings [28], e.g. for systems with variable spatial grids and local processes. In particular, Colored Petri Nets typically model systems where components are repeated in a symmetric manner, often without spatial differentiation. For instance, in a spatial model where different grid cells have distinct environmental conditions, infrastructure levels, or economic activities, Colored Petri Nets struggle to differentiate these spatial entities effectively. This is because they tend to assume the same internal parametrized structure of the Petri Net model, i.e. the places and transitions for each grid cell. Colored Petri Nets require uniformity in the structure and behavior of their components, which means they cannot naturally encode large-scale spatial grids where each unit has unique characteristics or interactions. Automated model assembly is therefore important for scaling beyond small-scale grids, such as for a handful of places representing whole countries [18]. The challenges of Petri Net modeling are further underscored by the plethora of diverse software tools available for Petri Net modeling.

[1]For all Google Scholar searches, the results are filtered to include publications since 2020 and were accessed on August 26, 2024. The numbers may vary as new articles continue to be published.

1.4. Petri Net software tools

There a number of published, open-source Petri Net simulation tools. Most rely on a graphical user interface (GUI) to construct and simulate Petri Net models, such as COSMOS [29], CPN Tools [30], PNetLab [31], ePNK [32], IOPT-Tools [33], WoPeD [34], ITS-Tools [35], and most recently Snoopy [36]. Other tools provide programming interfaces in languages such as MatLab [6,37] or Python [38]. However, many of these open-source tools are no longer under active development, or lack built-in features required to easily specify and modify large-scale systems. Additionally, none of the above have capabilities to run Petri Net models on a server, which can conserve user resources and accelerate the simulation process [5]. We therefore chose to focus on augmenting the Petri Net tool, Spike [5], which has been recognized for its efficiency and scalability [39–42]. Our software package is also designed in a modular way to enable future modifications to be compatible with other toolkits.

Spike is a command-line tool designed for continuous, stochastic, and hybrid simulation of Petri Nets [5]. It supports server-based operations, ensuring the reproducibility of models and configurations. Spike also supports Systems Biology Markup Language files [5], allowing for greater flexibility in modeling complex networks in a standard format for future integration and analysis. *Systems Biology Markup Language* (SBML) is a file format designed for representing computational models in a declarative manner, allowing for seamless exchange between different software systems [43]. The exchange allows for wider utility in a variety of domains, such as modeling cellular systems [44], biochemical reactions [45], disease modeling [46], and Boolean networks with Petri Nets [47] (Another, related markup language, Petri Net Markup Language (PNML), does not appear to have been supported since 2015 [48,49]). In addition, Spike has a command line feature that, unlike similar tools such as Snoopy [36], is fully functional, making it suitable for automating multiple simulation experiments. Because Spike is a tool in the PetriNuts software family,[2] users have access to a wealth of interoperable Petri Net analysis tools, such as Charlie [50] for analysis of Petri Net models by standard Petri Net theory and Marcie [51] for model checking and reachability analysis.[3]

However, to effectively utilize Spike, users must be proficient in manually modeling Petri Nets in either the SBML or Spike's internal format language, Abstract Net Description Language (ANDL). These formats are not commonly used or known outside of specialized scientific subfields and have their own intricacies and complexities. SBML moreover is based on XML, which is a widely used and mature standard but generally unfamiliar to students and researchers in the natural and social sciences. For a broader audience, then, there is clear value in facilitating the design of large-scale Petri Nets based on more commonly used file formats, such as shapefiles and CSV. A shapefile (.shp) is a spatial data format developed by Environmental Systems Research Institute, Inc. (ESRI) to store nontopological geometry and attribute information for the spatial features in a data set [52]. These file formats are particularly important for the integration and interoperability of

[2]We refer the reader to https://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/ Software.
[3]The integration of these software tools is currently out of the scope of this version of MPAT and is left as a possible future extension.

diverse datasets. To our knowledge, there is no assembly tool that can take heterogeneous information layers relevant to an individual Petri Net and give their internal interconnectivity assessment of large-scale Petri Net.

1.5. Contributions

Specifically, we present a Python-based package, MPAT, that facilitates the modular assembly of large-scale Petri nets, empowering users to model extensive spatial grids with greater efficiency and ease. Rather than creating another Petri Net simulator from scratch, this package expands the utility of the existing tool Spike. The aim of MPAT is to bridge the gap between the theoretical complexity of assembling Petri Net models and their practical application, making them more accessible and usable for a broader range of spatial modeling tasks. To our knowledge, no other software package provides these capabilities for the assembly of Petri Nets at this scale. The proposed toolkit for modular assembly of large-scale Petri Nets is shown in Fig. 1.

In particular, the contributions of this research are fourfold:

1. The proposed software toolkit takes a shapefile and generates CSV file describing a grid of the desired size of the spatial Petri Net model with adjacent patches (or grid cells) connected. This streamlines the process of creating and managing complex models without needing to manually configure each connection.

2. As another option, the software takes input from a CSV file of the adjacency matrix of the patches (or grid cells) and outputs the spatial structure of the Petri Net model, complete with interconnected places and transitions.

3. The proposed software can take in heterogeneous, individual information layers, such as vegetation, wind speed, humidity, etc. in a vector format. By integrating these diverse information layers, the software enables users to fine-tune Petri Net parameters at more granular spatio-temporal levels not currently available in existing Petri Net software tools.

4. The software streamlines efficient parameter search across heterogeneous parameter spaces by consolidating Petri Net results into a unified CSV file. This supports comprehensive analysis and comparison of various model configurations.

The remainder of this paper is structured as follows: Section 2 provides an introduction to the structure and features of MPAT. In Section 3, we present two use cases that demonstrate the validity and the utility of MPAT, while Section 4 describes the software's impact and broader implications. Finally, in Section 5, we share our insights into the new software toolkit and future directions in the concluding section.

2. Software description

In this section, we examine the proposed tool, MPAT. First, we outline the software architecture, followed by a detailed exploration of its functionality and the formats of its input and output.

2.1. Software architecture

MPAT has five primary Python source codes, specifically *Polygon.py, InfLayers.py, SIRModelSBML.py, HyperParameters.py, RunThroughSpike.py*, and *CSVFileReader.py*. Fig. 2 provides a flowchart of the high-level overview of MPAT. Below, we describe each function in Fig. 2. To start, *Polygon.py* manages the conversion of a shapefile into a GeoPandas dataframe, organizing grid cells into patches and linking them to their neighboring patches. Additionally, *InfLayers.py* handles the information layers in vector format. *SIRModelSBML.py* generates the corresponding Petri Net configuration model while *HyperParameters.py* handles the parameter space for the model. Next, *RunThroughSpike.py* handles the Petri Net simulations by running each configuration model through the Petri Net simulator, Spike [53]. To avoid any compatibility difficulties, the user sets the path to Spike. Finally, the output of the simulation results is handled by *CSVFileReader.py* to facilitate the accessibility of the results and an efficient way for analysis.

2.2. Software functionalities

In this section, we describe the major functionalities of MPAT (see Fig. 2). To begin, the user inputs the choice of a shapefile for a geographic area or the adjacency file as a CSV file of predefined patches. If it is a shapefile, *Polygon.py* generates a grid of patches of the user's desired size and outputs the GeoPandas data frame of the grid and adjacency matrix. On the other hand, if the adjacency matrix file is already defined by the user, then we proceed directly to the Petri Net modeling component of the toolkit. Additionally, the user input of information layers flows into *InfLayers.py* for linking with the grid of patches from *Polygon.py* or adjacency matrix. Given the adjacency matrix either from *Polygon.py* or the user, *SIRModelSBML.py* generates the Petri Net model with desired input parameters, such as the initial number of tokens/markings for each patch and the arc weights between the patches. For the initialization of one instance of the parameters, such as tokens/markings, the user can import a CSV file with the corresponding name of the place (transition) and the number of tokens/markings (arc weight). For places that are not specified, the default token/marking is zero, and the default arc weight is one for transitions. If no CSV is imported, then the default settings of 100 tokens/markings in the first place of the order of the Petri Net are assigned, and the default arc weight of the transitions is one. The output is the corresponding Petri Net model in XML/SBML and ANDL file formats. Generating the SBML and ANDL files uses Python's built-in xml.etree submodule. These files are the reference configuration files for parallelizable or multi-scale modeling for user-defined hyperspace of parameter values in *HyperParameters.py*. Given the Petri Net model files, *RunThroughSpike.py* runs each file through the Petri Net simulator, Spike. The output of *RunThroughSpike.py* is a collection of CSV files of the resulting simulation results for each simulation run. Because of the unwieldy number of CSV files produced by executing multiple runs, *CSVFileReader.py* compiles all of the CSV files into a single one, creating an automatic directory and providing a basic analysis of the results. Since the files are in CSV format, they can be exported to a variety of software applications by the user.

2.3. Input and output formats

MPAT has three primary input options. The first option is a shapefile of the geographic region. A shapefile, often denoted by the common extension .shp, serves as the standard format for representing spatial data or features, encompassing polygons. In the shapefile, the polygon is a fundamental unit of spatial representation. The second input option is a CSV file of the adjacency matrix of patches (or grid cells). The third is an optional CSV file of initialized parameters, such as tokens/markings and arc weights, for one instance of the Petri Net model.

An intermediate output is the corresponding Petri Net model in ANDL (.andl) and SBML (.xml) files. After executing the model runs in the Petri Net simulator, Spike, MPAT compiles the collection of CSV files into a merged CSV output describing the Petri Net simulation results for analysis and visualization. We now proceed to illustrate the functionality and utility of MPAT.

3. Illustrative examples

In this section, we provide two examples to demonstrate the utility of the proposed tool. The first is validation with a percolation model describing the spreading of forest fires. The second demonstrates the assembly capabilities of the tool in the context of epidemiological models for United States counties.[4] These examples provide accessible interpretation for an interdisciplinary audience for validation of the code. In particular, the percolation model of fire propagation has mathematical verification [54–56], while an epidemiological model is widely used across various academic domains, such as in public health [57] and insurance [58]. Indeed, users may consider more complex models using Petri Nets. The examples are found under the examples folder in the MPAT GitHub repository.

3.1. Validation with a percolation model of forest fire spreading

In this example, we validate the results of the MPAT tool with those of a percolation model on a two-dimensional square lattice using the Moore neighborhood (i.e., chess queen adjacency between different patches). For motivation, consider the dynamics of a forest fire spreading through a landscape. Here, trees are represented by patches on the lattice, and the fire spreads from one tree to its neighboring trees according to the rules of percolation theory.[5] Specifically, each patch has a tree that is either alive (white patch) or on fire (red patch) (see Fig. 3).[6] Furthermore, the critical density (denoted as p_c) is the probability at which there is a continuous path of patches with trees on fire from one side of the lattice to the opposite side. Below this critical density, the fire has a low probability of dying out because there is not a sufficient connected path of patches with trees on fire. At or above

[4]MPAT can also work with any country or user's defined shapefile.
[5]For a comprehensive introduction to percolation theory, we refer the reader to Lanchier [59].
[6]For simplicity, we assume that each patch has only one tree.

this density, the fire can percolate through the entire lattice. In particular, at a critical density of $p_c \approx 0.41$, the fire spreading behavior shifts from an extinction regime, where the fire eventually dies out, to an uncontrollable regime, where the fire continues to spread, forming large clusters of burning trees [54–56] (see the blue path in Fig. 3). This critical point marks the transition between different behaviors of the system, demonstrating the importance of understanding percolation thresholds in predicting and managing the spread of forest fires. By comparing the MPAT tool's results with those of the established percolation model, we can validate the tool's accuracy in simulating such large-scale, spatial phenomena.

In particular, Fig. 4 compares the Petri Net model from MPAT with percolation model simulations, demonstrating the percolation threshold at $p_c \approx 0.41$. This result aligns with established findings in the mathematical literature [54–56]. Additionally, Fig. 4(b) illustrates the mean cluster size for different occupation probabilities, further validating MPAT.

3.2. Large-scale, multi-patch epidemiological models

To illustrate the capabilities of MPAT for building large-scale spatial Petri Net models, we construct epidemiological models spanning all 3066 counties in the contiguous United States. These models account for the interconnections with neighboring counties, utilizing the adjacency file from the U.S. Census Bureau [60] (see Fig. 5). For simplicity, we consider a standard Susceptible (S), Infected (I), and Recovered (R) model as a Petri Net with transitions between neighboring S and I counties.[7] Next, MPAT generates the corresponding SBML and ANDL files, which are then implemented in Spike. The output is a unified CSV file of simulations with different stylized parameters. The user can alter the initial parameters as desired for multiple scenarios. In terms of computing resources, the simulations were conducted on an Intel Xeon Gold 6226R CPU @ 2.90 GHz with 64 threads and 512 GB of memory on a Linux server. The resulting runtime was approximately 2 h and 7 min.

4. Impact

The existing Petri Net simulation tools often rely on a GUI, making it cumbersome for large-scale assembly. For the tools that use a command-line interface, there is a critical lack of modular assembly of Petri Net models both at scale and in overcoming the specific language of the tool and Petri Net modeling structure.

In addressing these challenges, MPAT is a Python-based modular assembly toolkit for Petri Net modeling. This development not only makes Petri Net modeling more accessible but also unlocks additional potential for Petri Net simulators, such as Spike. The MPAT toolkit allows for seamless integration of various data sources, information layers, and modeling requirements. By automatically generating a grid from shapefiles and establishing connections between adjacent patches, it simplifies the creation and management of complex spatial Petri Net models. The ability to input an adjacency matrix from a CSV file further enhances the flexibility and ease of use, enabling users to quickly define and visualize

[7] For consistency, we construct the epidemiological models as Petri Nets following Connolly, Gilbert, and Heiner [18] and Segovia [19] for each county.

the spatial structure of their models. Additionally, MPAT streamlines the parameter search process across heterogeneous parameter spaces by consolidating results into a unified CSV file. This facilitates comprehensive analysis and comparison of different model configurations, making MPAT an invaluable tool for researchers and practitioners in the field of Petri Net modeling. MPAT enables users to process spatial data from various geographic regions more efficiently than ever before, paving the way for more sophisticated and granular analysis.

By leveraging the wide range utility of Python, MPAT makes complex and large-scale modeling of Petri Nets at varying spatial scales more manageable and user-friendly. The capabilities of MPAT using the SBML and ANDL formats allows for further extensions of this tool into the collection of Petri Net tools in the PetriNuts family of tools, such as Charlie [50], Patty [61], and Snoopy [36], as well as future integrations of machine learning algorithms for analysis (see e.g. Pinto et al. [62], and Hwang and Chang [63]).

In summary, MPAT is a powerful tool for the construction of large-scale, multi-patch Petri Nets for modular assembly. Rather than creating another software for Petri Net simulations, this tool expands the user-ability and modeling capabilities of an existing and commonly used Petri Net simulator, Spike. In doing so, MPAT helps reduce the learning curve of Spike but also broadens the use-cases of the modular assembly of Petri Nets into other areas, such as hydrology, management, and environmental science, where users may be unfamiliar with Petri Net modeling.

5. Conclusions

MPAT offers a user-friendly and flexible framework for the modular assembly of Petri Nets, paving the way for new research avenues of large-scale Petri Net modeling. The integration of heterogeneous vector data layers, such as population, vegetation, wind speed, and humidity, enhances modelers' abilities to parameterize Petri Net models at more granular spatio-temporal levels. By simplifying the user experience and boosting adaptability, the proposed software equips seasoned researchers and newcomers to Petri Nets with a new tool for large-scale spatial modeling and analysis.

Future work may encompass several extensions of the software. The first is to expand it to the community of hybrid Petri Net models, which incorporate combinations of continuous and stochastic places and transitions. The second is to expand the pipeline for analysis to allow for a wider range of machine-learning resources, such as *Scikit-learn* and *SciPy*.

Acknowledgment

This work was supported under the National Institutes of Health, United States grant DMS-1615879.

Data availability

Data will be made available on request.

References

[1]. Oliveira S, Leal AB, Teixeira M, Lopes YK. A classification of cybersecurity strategies in the context of discrete event systems. Annu Rev Control 2023;56:100907.

[2]. Peterson JL. Petri nets. ACM Comput Surv 1977;9(3):223–52.

[3]. Reisig W. Elements of distributed algorithms: modeling and analysis with Petri nets. Springer Science & Business Media; 1998.

[4]. Reisig W. Petri nets: an introduction, vol. 4, Springer Science & Business Media; 2012.

[5]. Chodak J, Heiner M. Spike–reproducible simulation experiments with configuration file branching. In: Computational methods in systems biology: 17th international conference, CMSB 2019, Trieste, Italy, September 18–20, 2019, proceedings 17. Springer; 2019, p. 315–21.

[6]. Davidrajuh R. Gpensim: a new Petri net simulator. InTech; 2010.

[7]. Kearney MR, Gillingham PK, Bramer I, Duffy JP, Maclean IM. A method for computing hourly, historical, terrain-corrected microclimate anywhere on earth. Methods Ecol Evol 2020;11(1):38–43.

[8]. Jiménez-Alfaro B, Chytrỳ M, Mucina L, Grace JB, Rejmánek M. Disentangling vegetation diversity from climate–energy and habitat heterogeneity for explaining animal geographic patterns. Ecol Evol 2016;6(5):1515–26. [PubMed: 26900451]

[9]. Lin L, Tang C, Liang Q, Wu Z, Wang X, Zhao S. Rapid urban flood risk mapping for data-scarce environments using social sensing and region-stable deep neural network. J Hydrol 2023;617:128758.

[10]. Jin Z, Fialko Y. Finite slip models of the 2019 Ridgecrest earthquake sequence constrained by space geodetic data and aftershock locations. Bull Seismol Soc Am 2020;110(4):1660–79.

[11]. Valente F, Laurini M. Tornado occurrences in the United States: a spatio-temporal point process approach. Econometrics 2020;8(2):25.

[12]. Sajjad M, Lin N, Chan JC. Spatial heterogeneities of current and future hurricane flood risk along the US Atlantic and Gulf coasts. Sci Total Environ 2020;713:136704. [PubMed: 32019039]

[13]. Gao L, Shi P. Leveraging high-resolution weather information to predict hail damage claims: A spatial point process for replicated point patterns. Insurance Math Econom 2022;107:161–79.

[14]. Soliman M, Newlands NK, Lyubchich V, Gel YR. Multivariate copula modeling for improving agricultural risk assessment under climate variability. Variance 2023;16(1).

[15]. Bucheli J, Conrad N, Wimmer S, Dalhaus T, Finger R. Weather insurance in European crop and horticulture production. Clim Risk Manag 2023;41:100525.

[16]. Boyle E, Inanlouganji A, Carvalhaes T, Jevtić P, Pedrielli G, Reddy TA. Social vulnerability and power loss mitigation: A case study of Puerto Rico. Int J Disaster Risk Reduct 2022;82:103357.

[17]. Chandrappa VY, Ray B, Ashwatha N, Shrestha P. Spatiotemporal modeling to predict soil moisture for sustainable smart irrigation. Internet Things 2023;21:100671.

[18]. Connolly S, Gilbert D, Heiner M. From epidemic to pandemic modelling. Front Syst Biol 2022;2:861562.

[19]. Segovia C. Petri nets in epidemiology. 2022, arXiv preprint arXiv:2206.03269.

[20]. Zaitsev DA, Shmeleva TR, Gizurarson S. Reenterable colored Petri net model of Ebola virus dynamics. In: 2023 IEEE international conference on machine learning and applied network technologies. IEEE; 2023, p. 1–6.

[21]. Yan R, Dunnett S, Andrews J. A Petri net model-based resilience analysis of nuclear power plants under the threat of natural hazards. Reliab Eng Syst Saf 2023;230:108979.

[22]. Kulagin V, Muraviev N. Software for modeling distributed systems using the Petri net apparatus. In: 2024 wave electronics and its application in information and telecommunication systems. IEEE; 2024, p. 1–5.

[23]. Capra L, Köhler-Bußmeier M. Modular rewritable Petri nets: An efficient model for dynamic distributed systems. Theoret Comput Sci 2024;990:114397.

[24]. Gilbert D, Heiner M, Liu F, Saunders N. Colouring space-a coloured framework for spatial modelling in systems biology. In: Application and theory of Petri nets and concurrency: 34th

international conference, PETRI NETS 2013, Milan, Italy, June 24–28, 2013. proceedings 34. Springer; 2013, p. 230–49.

[25]. Gao Q, Liu F, Tree D, Gilbert D. Multi-cell modelling using coloured Petri nets applied to planar cell polarity. In: Proceedings of the 2nd international workshop on biological processes & Petri nets, satellite event of PETRI NETS, vol. 724, Citeseer; 2011, p. 135–50.

[26]. Zaitsev DA, Shmeleva TR, Retschitzegger W. Spatial specification of grid structures by petri nets. In: Micro-electronics and telecommunication engineering: proceedings of 4th ICMETE 2020. Springer; 2021, p. 253–63.

[27]. Zaitsev DA, Shmeleva TR, Pröll B. Spatial specification of hypertorus interconnect by infinite and reenterable coloured Petri nets. Int J Parallel Emergent Distrib Syst 2022;37(1):1–21.

[28]. Carrasquel JC, Lomazova IA, Rivkin A. Modeling trading systems using Petri net extensions. In: PNSE@ Petri nets. 2020, p. 118–37.

[29]. Ballarini P, Barbot B, Duflot M, Haddad S, Pekergin N. HASL: A new approach for performance evaluation and model checking from concepts to experimentation. Perform Eval 2015;90:53–77.

[30]. Westergaard M, Kristensen LM. The access/cpn framework: A tool for interacting with the cpn tools simulator. In: International conference on applications and theory of Petri nets. Springer; 2009, p. 313–22.

[31]. Basile F, Carbone C, Chiacchio P. PNetLab: a tool for the simulation, analysis and control of discrete event systems based on petri nets. IFAC Proc Vol 2004;37(18):213–8.

[32]. Kindler E. The ePNK: an extensible Petri net tool for PNML. In: Applications and theory of Petri nets: 32nd international conference, PETRI NETS 2011, Newcastle, UK, June 20–24, 2011. proceedings 32. Springer; 2011, p. 318–27.

[33]. Gomes L, Moutinho F, Pereira F. IOPT-tools—A web based tool framework for embedded systems controller development using Petri nets. In: 2013 23rd international conference on field programmable logic and applications. IEEE; 2013, 1–1.

[34]. Freytag T. Woped–workflow petri net designer. 2005, p. 279–82, University of Cooperative Education.

[35]. Thierry-Mieg Y. Symbolic model-checking using ITS-tools. In: Tools and algorithms for the construction and analysis of systems: 21st international conference, TACAS 2015, held as part of the European joint conferences on theory and practice of software, ETAPS 2015, London, UK, April 11–18, 2015, proceedings 21. Springer; 2015, p. 231–7.

[36]. Heiner M, Herajy M, Liu F, Rohr C, Schwarick M. Snoopy–a unifying Petri net tool. In: Application and theory of Petri nets: 33rd international conference, PETRI NETS 2012, Hamburg, Germany, June 25–29, 2012. proceedings 33. Springer; 2012, p. 398–407.

[37]. Júlvez J, Mahulea C, Vázquez C-R. SimHPN: A MATLAB toolbox for simulation, analysis and design with hybrid Petri nets. Nonlinear Anal Hybrid Syst 2012;6(2):806–17.

[38]. Pommereau F. SNAKES: A flexible high-level petri nets library (tool paper). In: Application and theory of Petri nets and concurrency: 36th international conference, PETRI NETS 2015, Brussels, Belgium, June 21–26, 2015, proceedings 36. Springer; 2015, p. 254–65.

[39]. Schwarick M, Rohr C, Liu F, Assaf G, Chodak J, Heiner M. Efficient unfolding of coloured Petri nets using interval decision diagrams. In: Application and theory of Petri nets and concurrency: 41st international conference, PETRI NETS 2020, Paris, France, June 24–25, 2020, proceedings 41. Springer; 2020, p. 324–44.

[40]. Bilgram A, Jensen PG, Pedersen T, Srba J, Taankvist PH. Improvements in unfolding of colored petri nets. In: Reachability problems: 15th international conference, RP 2021, liverpool, UK, October 25–27, 2021, proceedings 15. Springer; 2021, p. 69–84.

[41]. Mahmoud EA, Herajy M, Ziedan IE, Shehata HI. Formal verification confirms the role of p53 protein in cell fate decision mechanism. Theory Biosci 2023;142(1):29–45. [PubMed: 36510032]

[42]. Bilgram A, Jensen PG, Pedersen T, Srba J, Taankvist PH. Methods for efficient unfolding of colored Petri nets. Fund Inform 2023;189.

[43]. Keating SM, Waltemath D, König M, Zhang F, Dräger A, Chaouiya C, Bergmann FT, Finney A, Gillespie CS, Helikar T, et al. SBML level 3: an extensible format for the exchange and reuse of biological models. Mol Syst Biol 2020;16(8):e9110. [PubMed: 32845085]

[44]. Pleyer J, Fleck C. Agent-based models in cellular systems. Front Phys 2023;10:968409.

[45]. Moretti S, Tran VDT, Mehl F, Ibberson M, Pagni M. MetaNetX/MNXref: unified namespace for metabolites and biochemical reactions in the context of metabolic models. Nucleic Acids Res 2021;49(D1):D570–4. [PubMed: 33156326]

[46]. Ostaszewski M, Niarakis A, Mazein A, Kuperstein I, Phair R, Orta-Resendiz A, Singh V, Aghamiri SS, Acencio ML, Glaab E, et al. COVID19 disease map, a computational knowledge repository of virus–host interaction mechanisms. Mol Syst Biol 2021;17(10):e10387. [PubMed: 34664389]

[47]. Trinh V-G, Benhamou B, Soliman S. Trap spaces of Boolean networks are conflict-free siphons of their Petri net encoding. Theoret Comput Sci 2023;971:114073.

[48]. Weber M, Kindler E. The petri net markup language. In: Petri net technology for communication-based systems: advances in Petri nets. Springer; 2003, p. 124–44.

[49]. Petri net markup language (PNML). 2015, https://www.pnml.org/, online; [Accessed 3 January 2023].

[50]. Heiner M, Schwarick M, Wegener J-T. Charlie–an extensible Petri net analysis tool. In: Application and theory of Petri nets and concurrency: 36th international conference, PETRI NETS 2015, Brussels, Belgium, June 21–26, 2015, proceedings 36. Springer; 2015, p. 200–11.

[51]. Heiner M, Rohr C, Schwarick M. MARCIE–model checking and reachability analysis done efficiently. In: Application and theory of Petri nets and concurrency: 34th international conference, PETRI NETS 2013, milan, Italy, June 24–28, 2013. proceedings 34. Springer; 2013, p. 389–99.

[52]. ESRI. Shapefile technical description. 1998, https://www.esri.com/content/dam/esrisites/sitecore-archive/Files/Pdfs/library/whitepapers/pdfs/shapefile.pdf, online; [Accessed 12 August 2024].

[53]. Chodak J, Heiner M. Spike-a command line tool for continuous, stochastic & hybrid simulation of (coloured) Petri nets. In: Proceedings of the workshop AWPN. 2018.

[54]. Dutta S, Sen S, Khatun T, Dutta T, Tarafdar S. Euler number and percolation threshold on a square lattice with diagonal connection probability and revisiting the island-mainland transition. Front Phys 2019;7:61.

[55]. Feng X, Deng Y, Blöte HW. Percolation transitions in two dimensions. Phys Rev E 2008;78(3):031136.

[56]. Perestrelo SA, Grácio MC, Ribeiro Nd, Lopes LM. A multi-scale network with percolation model to describe the spreading of forest fires. Mathematics 2022;10(4):588.

[57]. Kudryashov NA, Chmykhov MA, Vigdorowitsch M. Analytical features of the SIR model and their applications to COVID-19. Appl Math Model 2021;90:466–73. [PubMed: 33012957]

[58]. Lefevre C, Picard P, Simon M. Epidemic risk and insurance coverage. J Appl Probab 2017;54(1):286–303.

[59]. Lanchier N. Stochastic modeling. Springer; 2017.

[60]. Bureau UC. County adjacency file. 2023, https://www.census.gov/geographies/reference-files/2010/geo/county-adjacency.html, online; [Accessed 3 May 2023].

[61]. Schulz K. An extension of the snoopy software to process and manage Petri net animations [Bachelor thesis], BTU Cottbus, Dep. of CS; 2008.

[62]. Pinto J, Costa RS, Alexandre L, Ramos J, Oliveira R. SBML2HYB: a Python interface for SBML compatible hybrid modeling. Bioinformatics 2023;39(1):btad044. [PubMed: 36661327]

[63]. Hwang S, Chang M. Similarity-principle-based machine learning method for clinical trials and beyond. Stat Biopharm Res 2022;14(4):511–22.

Author Manuscript
Author Manuscript
Author Manuscript

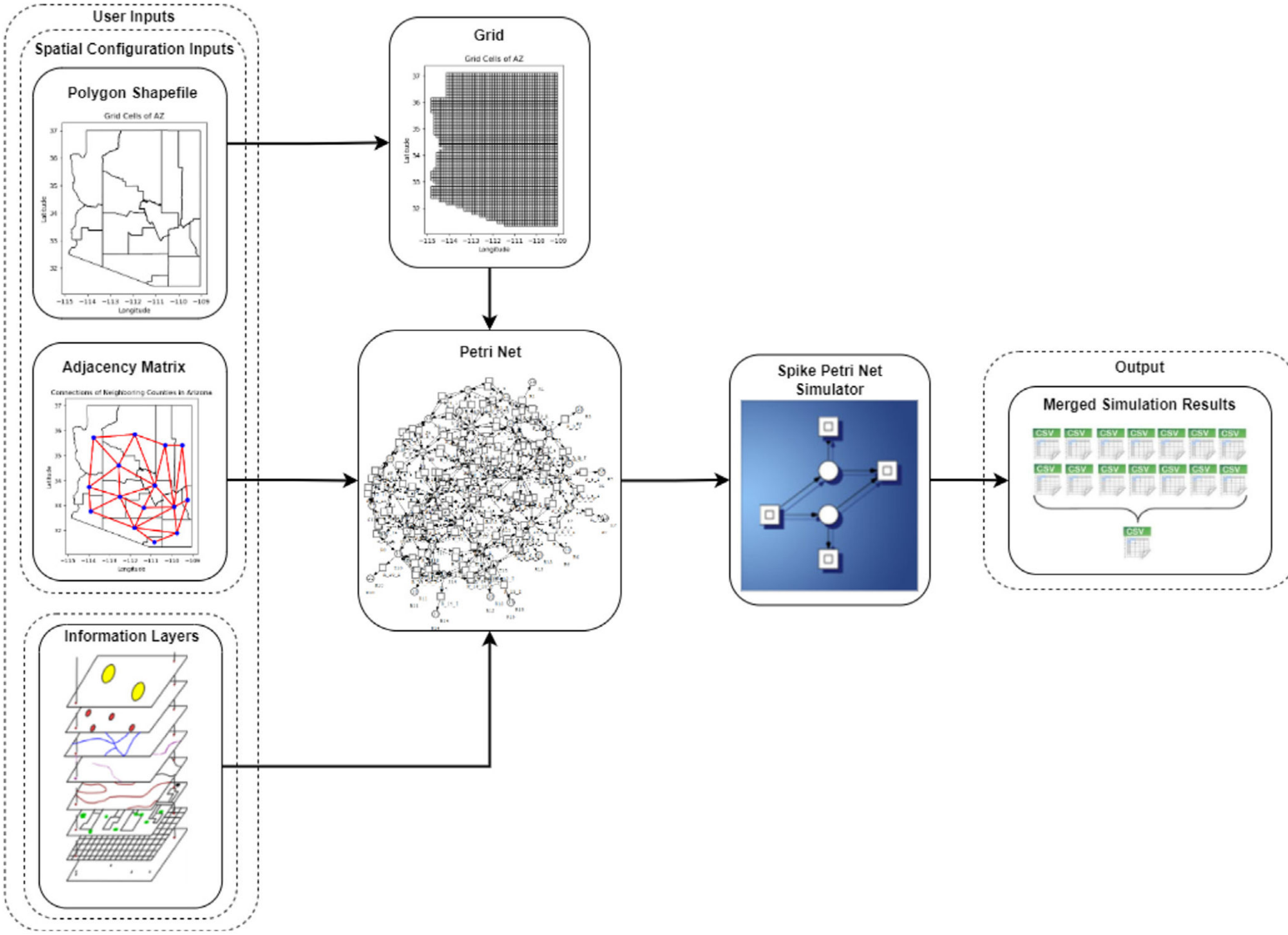


Fig. 1.
The proposed tool for modular assembly of large-scale Petri Nets.

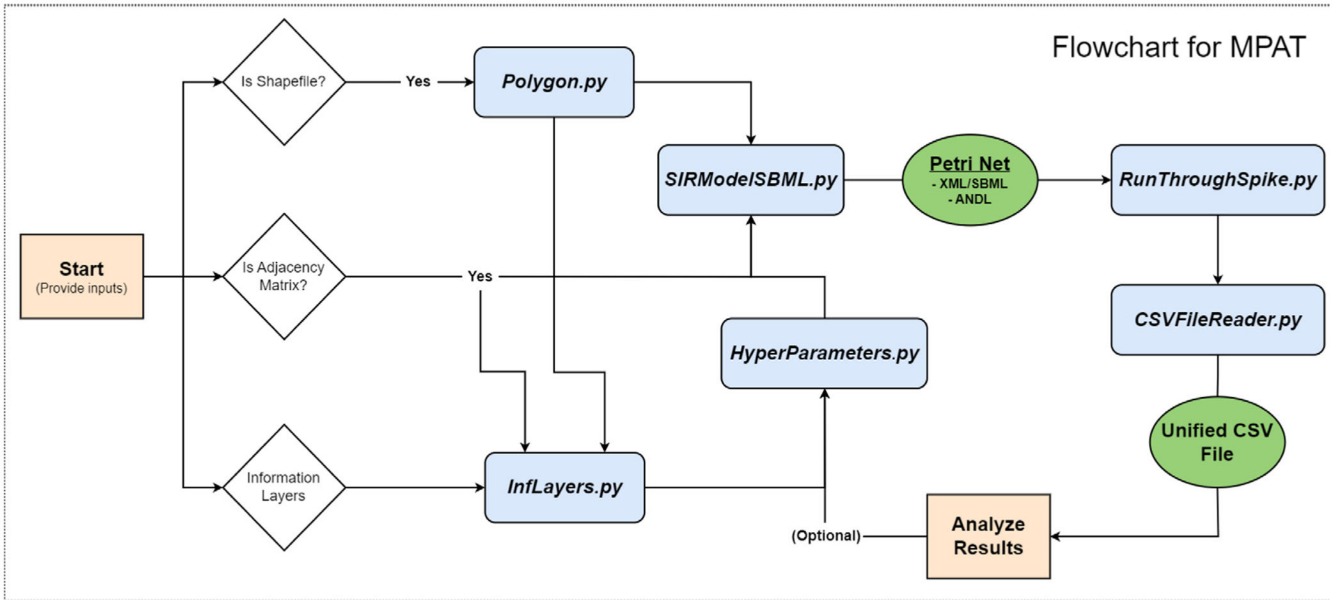


Fig. 2.

The flowchart of MPAT functions. The triangles indicate the initial input arguments. The blue rectangles indicate the Python functions and the green circles indicate the output files.

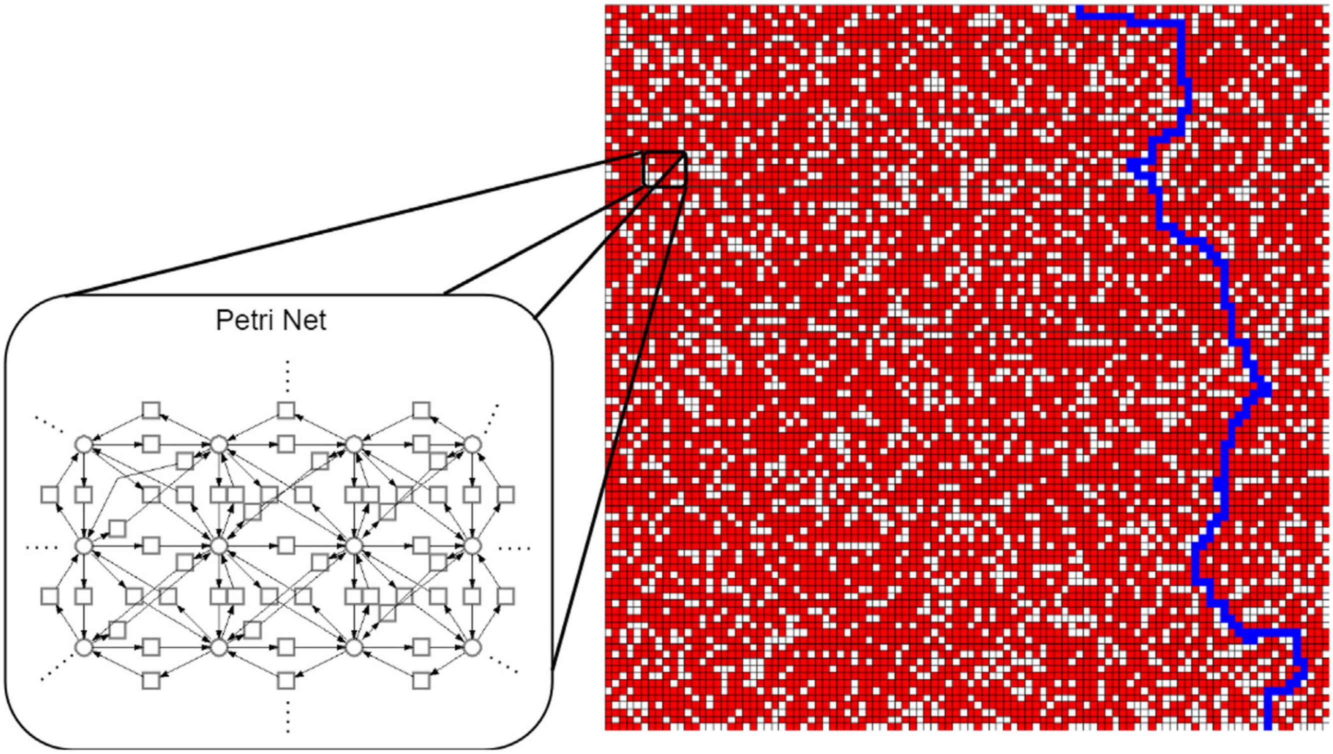
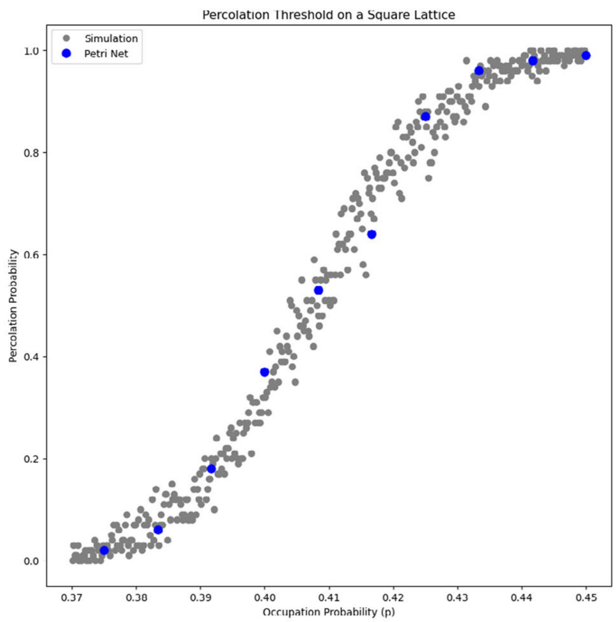
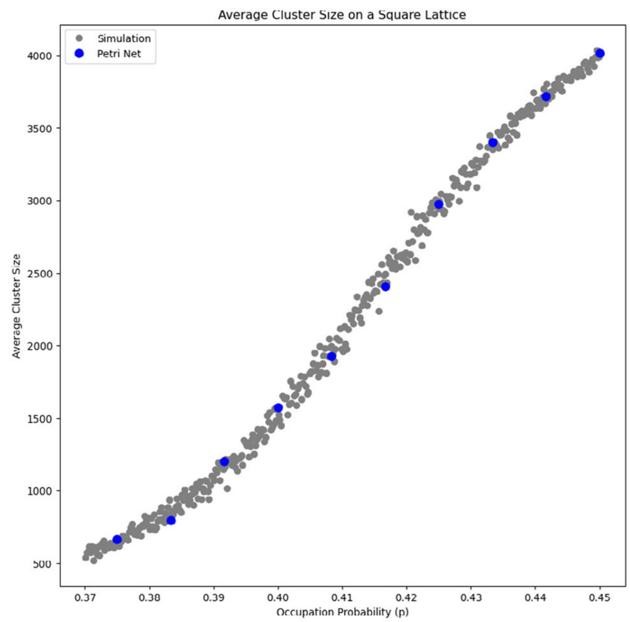


Fig. 3.

Visualization of one instance of the fire spread model with Petri Net representation on 100 × 100 lattice with probability $P = 0.41$ with Moore neighborhood. The white patches have the tree that is alive and the red patches have the tree on fire. The blue path is the percolation cluster.



(a)



(b)

Fig. 4.

Visualizations of simulations (gray) and Petri Net results (blue) for the (a) percolation threshold and (b) average cluster size on a 100 × 100 lattice with a Moore neighborhood. The percolation threshold, at which a phase transition is present, occurs at $p_c \approx 0.41$.

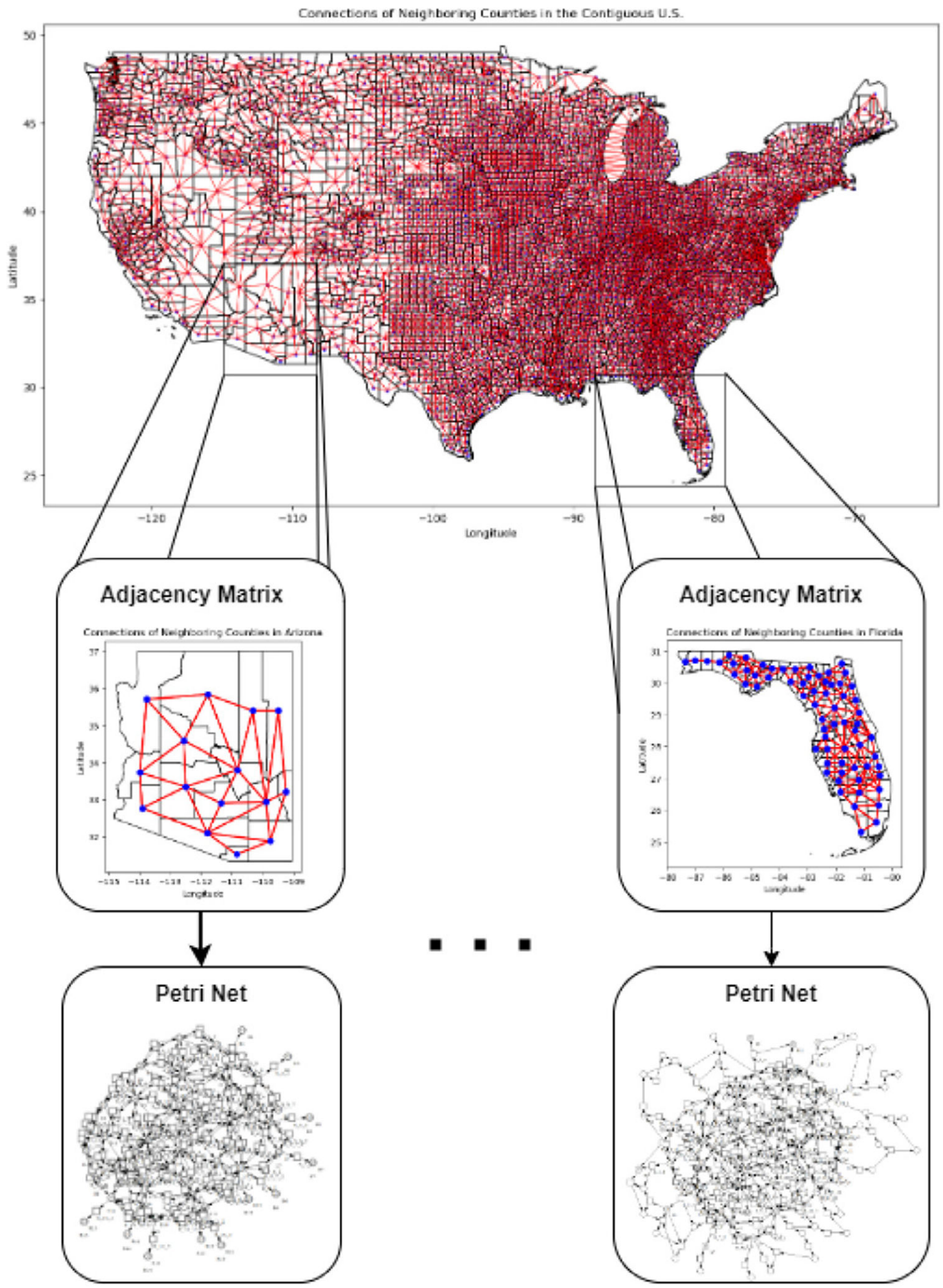


Fig. 5.
County level adjacency of the contiguous United States [8].

[8]According to the source file for county adjacency [60], "in some instances, the boundary between two counties may fall within a body of water, so it seems as if the two counties do not physically touch. These counties are included on the list as neighbors". This explains instances where counties are classified as neighbors near bodies of water.