Journal of Computational Science 81 (2024) 102371

Contents lists available at ScienceDirect

Journal of Computational Science

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

Flee 3: Flexible agent-based simulation for forced migration

Maziar Ghorbani [a], Diana Suleimenova [a], Alireza Jahani [a], Arindam Saha [a], Yani Xue [a], Kate Mintram [a], Anastasia Anagnostou [a], Auke Tas [b], William Low [b], Simon J.E. Taylor [a], Derek Groen [a,*]

[a] Modeling & Simulation Group, Department of Computer Science, Brunel University London, UK
[b] Save the Children International, London, UK

ARTICLE INFO

Keywords:
Conflict-driven displacement
Human migration
Emergency response support
Agent-based modelling
Parallel computing

ABSTRACT

Forced migration is a major humanitarian challenge today, with over 100 million people forcibly displaced due to conflicts, violence and other adverse events. The accurate forecasting of migration patterns helps humanitarian organisations to plan an effective humanitarian response in times of crisis, or to estimate the impact of possible conflict and/or intervention scenarios. While existing models are capable of providing such forecasts, they are strongly geared towards forecasting headline arrival numbers and lack the flexibility to explore migration patterns for specific groups, such as children or persons of a specific ethnicity or religion. Within this paper we present Flee 3, an agent-based simulation tool that aims to deliver migration forecasts in a more detailed, flexible and reconfigurable manner. The tool introduces adaptable rules for agent movement and creation, along with a more refined model that flexibly supports factors like food security, ethnicity, religion, gender and/or age. These improvements help broaden the applicability of the code, enabling us to begin building models for internal displacement and non-conflict-driven migration. We validate Flee 3 by applying it to ten historical conflicts in Asia and Africa and comparing our results with UNHCR refugee data. Our validation results show that the code achieves a validation error (averaged relative difference) of less than 0.6 in all cases, i.e. correctly forecasting over 70% of refugee arrivals, which is superior to its predecessor in all but one case. In addition, by exploiting the parallelised simulation code, we are able to simulate migration from a large scale conflict (Ukraine 2022) in less than an hour and with 80% parallel efficiency using 512 cores per run. To showcase the relevance of Flee to practitioners, we present two use cases: one involving an international migration research project and one involving an international NGO. Flee 3 is available at https://github.com/djgroen/flee/releases/tag/v3.1 and documented on https://flee.readthedocs.io.

1. Introduction

Forced migration is one of the most pressing humanitarian crises globally, with over 100 million forcible displaced due to conflicts, violence and other adverse events [1]. This includes persons displaced abroad (e.g., recognised, and unrecognised refugees) as well as internally displaced persons (IDPs). The factors that lead to forced migration are complex and frequently include the presence of armed conflicts, political instability, environmental disasters, and/or socioeconomic disparities. In addition, forced migration has far-reaching consequences, affecting not only the displaced persons themselves but also the host communities and regions.

Forecasting migration is important, because it can contribute insights to policymakers in at least three diverse ways [2]. First, policymakers, aid organisations, and researchers may use forecasts to formulate more effective strategies for managing and responding to conflict situations, including resource allocation, humanitarian assistance programmes for asylum seekers/unrecognised refugees, and the establishment of safe zones and camps. Second, forecasts may become useful in estimating the population sizes of displaced people in locations where empirical research is unsafe or impractical. And third, forecasting tools can potentially contribute to retrospectively analyse the impact of major interventions in historical refugee situations. If accurate and systematically applied, forecasting tools enable stakeholders to anticipate and mitigate potential challenges associated with mass migrations, such as the strain on host communities, resource scarcity, and the risk of disease outbreaks.

Computational modelling is a widely used technique that helps us to produce such forecasts and gain insight into the dynamics of forced migration. While various techniques have been employed, the current dominant approaches are machine learning (ML) based models [3,4]

* Corresponding author.
E-mail address: derek.groen@brunel.ac.uk (D. Groen).

https://doi.org/10.1016/j.jocs.2024.102371
Received 26 January 2024; Received in revised form 21 June 2024; Accepted 21 June 2024
Available online 26 June 2024

and Agent-Based Models (ABMs). Here, ML based models are trained on recent forecasting data (and related variables) to produce near-term forecasts, while ABMs focus on curating an understanding on how individual agents behave and interact within their ecosystem, with an indirect aim of producing thoroughly substantiated migration forecasts.

In general, ABMs have become popular due to their ability to model complex systems in a manner that is both realistic and relatable for stakeholders [5–9]. They do not need to rely on training data, like ML based approaches do, and offer the ability to explore behaviours of individuals, their effects on the ecosystem and emergent collective behaviours in an explainable way. While ML based models perform well in data-rich environments, ABMs are better suited for environments where the quality and quantity of available data is limited, or where fine-grained interventions or variations need to be introduced. In the case of forced migration, the observational data in the most severe humanitarian situations is often very limited in scale and quality, and fine-grained interventions often need to be modelled to account for road closures or the different ways that a conflict may evolve. Because of these aspects, earlier versions of the Flee simulation code presented here where adopted by several academic and humanitarian research groups.

However, the adoption of Flee 1 and Flee 2 for these purposes led to the rise of new challenges: while the generalised approach presented by Suleimenova et al. [10] allowed relatively rapid reuse of the code in new contexts with reasonable accuracy, the ability to incorporate context-specific knowledge into the simulations was limited by a relatively rigid conceptual model. This limitation led to spin-off developments for instance to develop support for migration driven by food insecurity [11] or strongly affected by weather factors [12], and also limited the applicability of Flee to forecast internal displacement.

Within this paper we present Flee 3, a flexible platform for simulating forced migration for a variety of contexts, including internal displacement and migration driven by factors other than conflict. Flee 3 also allows users to produce forecasts stratified by demographic characteristics such age, gender, religion or ethnicity. In Section 2 we present related work and summarise the history of Flee, while in Section 3 we provide a detailed explanation of the key conceptual and technical aspects in Flee 3. In Section 4 we describe the setup for our performance and accuracy tests, while in Section 5 we present the main results and explain the relevance of Flee to practitioners in this context. We provide a summary and key conclusions in Section 6.

2. Background and related work

2.1. Migration modelling and simulation

Simulation development involves the initial formulation of real-world problems, their translation into model representations, the conversion of models into computerised simulations, and the subsequent execution of experimental runs coupled with a comprehensive analysis of the outcomes. This approach, combined with a rigorous regime of verification, validation, and uncertainty quantification [13] helps ensure the accuracy and reliability needed to model complex phenomena [14]. To make a meaningful contribution to humanitarian efforts, migration modelling must adhere to these fundamental principles whilst respecting the rights of refugees and their associated data [15].

In the context of simulation, ABS offers several advantages over other approaches commonly used in migration modelling, such as artificial intelligence (AI) and ML based methods [16]. In particular, AI and ML approaches rely on historical reference data, which can be problematic in a humanitarian context where historical data is often incomplete or inconsistent. ABS allows developers to define behaviours on the individual level using qualitative evidence (as well as quantitative evidence), delivering complex behaviour through the interaction and aggregation of individual agents and their interactions with their local environment. In this context, it is important to validly define behaviours on the individual level, e.g. by obtaining evidence of such behaviours through observational reports or qualitative (or quantitative) survey data. As a result, ABS can be applied to model real-world systems that are subject to major data limitations or omissions.

ABS is also by definition well-suited for capturing intricate interactions and behaviours among individual agents within dynamic systems because the behaviours of individuals are explicitly defined. It has a proven track record for accurately representing complex population phenomena such as pedestrian dynamics [17] and land-use population displacement scenarios [18]. It is possible to model forced migration situations (or important aspects of it) using System Dynamics [19], Discrete Event Simulation (DES) [20], Cellular Automata (CA) [21], network models [22], Markov models [23], and game theory models [24]. However, in our case ABS allows us to easily introduce fine-grained interventions due to its explicit spatial representations, and to update behavioural rules directly and instantly due to its explicit mapping of humans to computational objects.

2.2. History of flee

Flee was initially developed to produce forecasts where conflict-displaced persons would cross the border and find safety. At the time it was new because such validated forecasting tools had not previously been published in the literature. Flee was first validated against refugee data from the Mali 2012 conflict [2] and later generalised and validated against two additional conflicts [10]. As part of the generalisation work, we introduced the notion of the Simulation Development Approach, or *SDA*, which is a specification of all the work required to develop and apply Flee in an emergency anticipation or response context. The SDA concept is applicable to a range of emergency forecasting use cases, as outlined by Groen et al. [25].

Even as other models emerged [3,4], Flee stands out as it uses a heuristics-based ABM approach that can be adapted to various conflict contexts without requiring retraining or recalibration. In terms of validation, Flee 1 and 2 correctly predicted 70% of the refugee arrivals across four different historical validation contexts, each run over a period of at least 300 days [10,26]. To make such validations conveniently possible, the code was linked up with an automation plugin, FabFlee [27], which used the FabSim3 automation toolkit to streamline remote HPC access and data curation, as well as establish shortcuts to perform systematic validation, uncertainty quantification and benchmarking.

The second version of Flee contains relatively minor adjustments in terms of agent behaviour [28], but included a parallel implementation [29,30] that enables users to run the code efficiently using up to 8192 cores for large-scale problems. This parallel performance is required to enable simulations with 10,000,000 or more agents (required for modelling displacement for large countries such as Ukraine or India), and to enable runs where agents have a high awareness of their spatial surroundings. Examples of the latter include the navigation of agents over grid-based location graphs, which at time of writing are in testing with Save the Children.

The philosophy of Flee 3 differs from its predecessors in that it allows for more forecasts stratified by demographic properties, and for more flexibility when defining the heuristics-based rule sets for agent spawning and agent movement. Many of these code features have been developed through extended collaborations with Save The Children, Columbia University [31], and IT Tools and Methods for Managing Migration Flows [32] consortium. In addition, due to this wider uptake, we are also able to present much more systematic validation results, showcasing how well a unified Flee rule set can be used to deliver accurate population displacement forecasts for a wide range of historical conflicts.

2.3. Agent-based migration simulation

In this subsection, we provide an overview of related work in the field of Agent-based Migration Simulation, with a focus on efforts that are less directly related to the Flee code, which we have already covered as part of the introduction.

Before the systematic adoption of ABM in migration research, [33] explored its use in studying householder migration and residential distribution dynamics. They introduced the concept of Multi-Agent Systems and simulated the decision-making processes of individual agents like households, within a spatial and temporal context. To our knowledge, [34] were the first to present a compelling case for using ABM in humanitarian aid. Their pioneering ABM, aimed at simulating policies to support refugee health and safety, showcased the method's effectiveness in capturing the complex interactions within refugee communities. Through sensitivity analysis and comparisons with system dynamics models, they highlighted the significance of policy simulations in refining strategies for refugee support. [35] presented an ABM, designed to capture population displacement in the Syrian city of Aleppo during the Syrian crisis in 2013. Their ABS established a computational perspective for anticipating and assessing future displacement in forced displacement scenarios. After 2015, research in the field of forced migration has further evolved to include advanced and validated modelling approaches, and uncovering new insights into the complex dynamics that underlie population displacement. In particular, the range of working and validated forced migration models has greatly increased. For instance, the UNHCR Jetson project [3], trialled machine-learning tools to perform forced displacement forecasts while [4] focused on predicting the destinations of internally displaced persons in the conflict-affected regions of Syria and Yemen. In addition, the ITFLOWS project has led to the development of ML-based models for bilateral forced migration [36,37]. In addition to ML models, the community has produced an impressive range of ABMs that complement the scope of Flee in important ways. [38] developed the CofMMA model, which specialises in simulating conflict-induced migration in areas where important data sources are unreliable or incomplete, while [39] developed a model that focuses strongly on the effects of migration on urban dynamics. Frydenlund et al. [40] initially developed an ABS that focused on forecasting the creation of spontaneous settlements by IDPs and followed this up with a different ABS for conflict-driven migration, combined with a comprehensive review of existing migration models [41]. Related to migration but focusing on a different area, [42] introduced an ABM focusing on characterising the well-being of asylum-seeking refugees in the Netherlands, while [43] examined the effects of different immigration policies.

Modelling humanitarian logistics in the context of forced migration is a subtopic that has recently gained attention in the field. Boshuijzen-van Burken et al. [44],Carammia et al. [45] have both developed models in this area, incorporating elements of value-sensitive design into their approaches. Within our group, we have also explored the placement of refugee camps as a facility location problem. Initially, we approached this through serious video games [46] to aid decision-making on refugee aid deployment policies. Subsequently, we adopted a more effective strategy by using a multi-objective simulation optimisation approach [47]. This research direction has also been investigated by Bayraktar et al. [48], who defined it as a multi-period mobile facility location problem with mobile demand and showcased their approach to the Honduras Migration Caravan Crisis in 2018.

On the theoretical side, there are also several studies directly relevant to our work. Klabunde and Willekens [49] provide insights into migration decision-making that can be integrated as behavioural theories into ABMs such as Flee. Thober et al. [50] highlight the importance of comprehensive ABMs, a consideration that aligns with Flee's goal of capturing complex interactions in migration processes while [51] systematic review underscores the significance of robust data collection, interdisciplinary collaboration, and rigorous model analysis. De Luca et al. [52] explore hybrid modelling approaches, reflecting a territory that remains largely uncharted in migration policies. Abel et al. [53] delve into the relationship between climate change, conflict, and forced migration and emphasise the need to consider context-specific dynamics rather than seeking a universal link. In addition, Pham and Luengo-Oroz [54] introduce a model-agnostic framework that attempts to bridge the gap between theory and practice in computational methods for addressing forced displacement challenges. Lastly, the work of Adib Bencherif and Stockemer [55] provides a good example of a solid empirical study that can be of direct use for model construction, in this case, the Malian civil war.

3. Overview of flee

3.1. Simulation development approach

A Simulation Development Approach is a systematic representation of the work required to develop new simulations for a specific purpose, e.g. for validating a simulation code or delivering a robust forecast. We first introduced the SDA concept as part of our initial Flee paper [10] and have subsequently generalised the concept for broader use in emergency response simulation building [25], advocating its uptake to improve emergency preparedness and response. Simulation Development for Flee has specific characteristics in that it (i) tends to be triggered at short notice due to an erupting or impending conflict event, (ii) relies on configuring an existing model (Flee) for a diverse set of speculative scenarios (e.g. different ways how a conflict may evolve [56]) and (iii) requires completion on a timescale of two weeks or less to be of use in a crisis situation.

Indeed in an emergency response context, one of the main contexts for which Flee is intended, it is important that simulations can be swiftly adapted to evolving conditions and are deployed for informing the humanitarian response in forced displacement situations. The SDA used for previous versions of Flee covers those important aspects, but is less well suited for Flee 3 because it does not incorporate the use and analysis of essential demographic information, nor the definition and analysis of the wider range of QoIs supported by Flee 3. Furthermore, the explicit support for alternative scenarios has been a frequently requested extension by NGOs such as Save the Children.

For Flee 3, we present the associated SDA in Fig. 1. This SDA overview includes the tasks required to validate a Flee simulation against real-world data as well as the tasks needed to produce a detailed and robust forecast. This SDA consists of six distinct phases which are normally performed in order to develop a working and relevant simulation that produces robust and actionable results. These phases include: (i) *Select Situation*, where the specific scenario or context for the simulation is identified and chosen; (ii) *Obtain Data*, which involves gathering the necessary data for building the simulation model; (iii) *Construct Model*, which entails developing the initial version of the simulation model based on the selected situation and the data obtained; (iv) *Refine Model*, where the model is iteratively improved by incorporating additional data, feedback, or new insights; (v) *Execute Simulation*, in which we execute simulations to generate the main results, including results that quantify aleatoric and epistemic uncertainties of a forecast or validation, report on key parameter sensitivities and incorporate alternative scenarios where a conflict or humanitarian intervention may differ from the baseline; And lastly (vi) *Analysis*, which involves interpreting these results and drawing conclusions.

In the first phase, *Select Situation*, the developer specified the location and time period of the simulation that will be developed. The time period will be historical in the case of validation or retrospective analysis, in the future in the case of a forecast, or encompass both historical and future dates in the case where a conflict is ongoing, and single simulation is used both for validation and forecasting. While the initial version of Flee was specifically used to forecast refugee arrivals over time in individual camps, Flee 3 caters to a broader range of forecasting

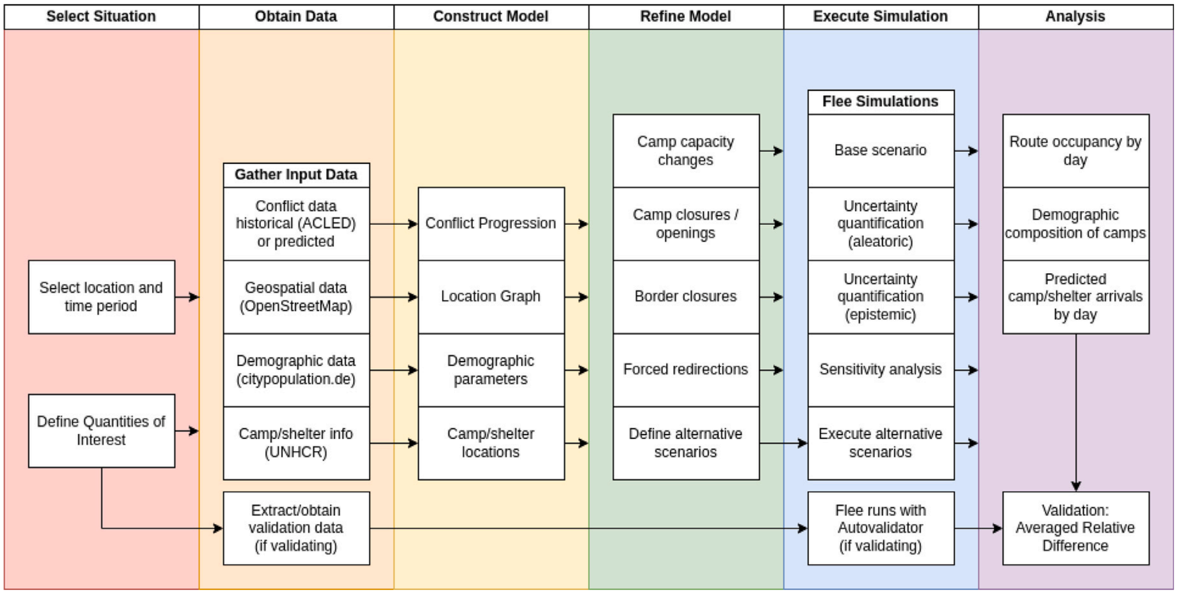


Fig. 1. Simulation Development Approach (SDA) diagram for developing simulations with Flee 3. Here, the bottom row tasks apply exclusively to a validation context, while all other tasks apply to both validation and forecasting context. Arrows indicate tasks that inform subsequent stages. Colour usage is consistent with prior publications for continuity, while more information about ACLED is provided in Section 4.1.

and analysis scenarios. As a result, it is important for the developer to also define their *Quantities of Interest (QoIs)*. For instance, users might be interested in the gender distribution of the arriving individuals, the most commonly used travel routes, or the likely destinations for persons displaced from a specific city. The selection of *QoIs* play a critical role in shaping the simulation's development, influencing aspects like data collection and scenario definition. It is therefore essential to establish this in the first phase of the SDA.

In the *Obtain Data* phase, we acquire the data that is necessary to construct the Flee simulation and (if needed) to validate it. This includes obtaining relevant conflict data, which can either be historical acquired from e.g. ACLED [57] or the Uppsala Conflict Database Program [58], a forecast made using a conflict model [56] or a forecast that matches conflict scenarios described by humanitarian experts. Flee 3 simulations also require geospatial data (including major settlements and their interconnecting routes) for constructing the location graph, as well as demographic data (e.g. population sizes, gender/age/religion composition of population) for defining accurate populations of displaced persons and information on e.g. camps and shelter locations to define possible destinations in the model. The collection of demographic data is particularly relevant as Flee 3 enables the assignment of (static) demographic attributes to agents, and allows for agent behaviours to be modified based on demographic attributes such as age or religion.

The Construct *Model phase* is dedicated to the development of the base Flee 3 model, using the data obtained in the previous phase. This includes defining the conflict progression as well as the location graph used in the Flee 3 simulation. We also introduce files describing the demographic makeup of populations in conflict areas (new in Flee 3), which are used as base probability distributions to define the demographic attributes of newly spawned displaced person agents in the simulation. Lastly, in this phase the developer needs to define the various camps and shelter locations in the model.

In the *Refine Model* phase, developers adapt the base Flee 3 model to ensure it accurately represents the expected humanitarian situation that it seeks to model. This includes modifying the simulation to account for (i) changes in camp capacity over time, (ii) openings of new camps or closures of existing camps, (iii) border closures, (iv) forced redirection policies, where authorities guide refugees to alternative locations, and (v) alternative conflict or intervention scenarios. This last task, though not included in the original Flee SDA, is particularly important both because humanitarian organisations often decide between different types of interventions, and because conflicts are unpredictable in nature.

Once the model has been refined, the developer is ready to perform the necessary simulation runs as part of the *Execute Simulation* phase. This phase may include (i) base scenario runs, (ii) replication runs to account for aleatoric uncertainty, (iii) additional runs to account for epistemic uncertainty (uncertainty in the value of input parameters), (iv) additional runs required to perform sensitivity analysis for key input parameters and (v) runs that account for alternative conflict and/or intervention scenarios. It is not uncommon for Flee forecasts to require runs for 5–10 different scenarios, each requiring 10 replications for aleatoric uncertainty and 100 replications for epistemic uncertainty, adding up to around 500 to 1000 runs for a single forecast. Sensitivity analysis tends to be performed as a separate study, using around 1000 to 10000 runs that are dynamically sampled from the input parameter space using tools such as EasyVVUQ [59]. It should be noted that runs performed as part of sensitivity analysis also help to quantify epistemic uncertainty for a specific scenario. As a result, when the number of scenarios is very small one would perform a sensitivity analysis, while for a large number of scenarios one would run smaller ensembles to account for epistemic uncertainty at least to some extent. Specifically in the case of code validation it is also possible to run Flee with the autovalidator tool that is build into FabFlee. This tool automatically executes Flee in the context of the conflict to be validated against, and automatically performs the necessary analysis to produce the validation results for the next phase.

Lastly, during in the *Analysis* step developers use the simulation results to extract their QoIs. These QoIs may include (i) the number of persons on given routes during given days, (ii) expected demographic composition of camp populations, (iii) camp/shelter arrivals by day or (iv) the averaged relative difference (ARD), if validation data is available. The FabFlee tool offers functions that allow users to obtain camp arrivals and ARD, while the first two QoIs are usually obtained by post-processing the agent and link log files.

3.2. Architecture

Flee 3 uses an object-oriented approach to model the migratory movements of individual persons in a spatial environment represented by a location graph. Whilst the choice for an object-oriented approach can result in inferior execution performance (compared to e.g. tensor-based methods [60]), it also delivers two more substantial benefits. First, it provides a more direct and intuitive mapping between the real world environment and its digital representation, reducing the model's cognitive complexity and the effort required to modify algorithms and the implementation as a whole. And second, the object-oriented approach also makes it easier to rapidly introduce targeted interventions, such as new movement rules for agents for specific demographics or the closure of individual borders.

We provide a high-level overview of the Flee 3 architecture in Fig. 2. Within this architecture, the Flee3 Ecosystem is a critical component that stores key elements of the Flee simulation state (see Fig. 4 for details). In Flee 3, most input files are processed by a dedicated component, whereas the Simulation settings file (and its accompanying reader) is used directly in the Ecosystem and in the definition of agent behaviours. The Simulation run scripts create an Ecosystem, which then manages the state of the Flee simulation and outputs diagnostic information on a periodic basis. The Flee3 Ecosystem propagates the simulation for each simulated day by invoking its `evolve` method to orchestrate agent movements across the location graph. This function relies on Agent behaviour definitions for spawning new agents, for calculating agent-specific location attractiveness scores and for calculating preferred routes for each agent.

Flee 3 supports two diagnostic systems. Firstly, it supports basic diagnostic outputs, such as population numbers in camps and rudimentary validation scores, through the `out.csv` file like previous versions. Second, Flee 3 supports detailed logs of agent locations (`agents.log`) and link populations (`links.log`), both recorded for each time step. Link log populations can be stratified by attributes such as age, gender, and ethnicity. Lastly, Flee 3 supports more advanced conflict dynamics through a `conflicts.csv` input file. This file details the locations, dates, and intensities of conflict events, a feature not supported in previous versions of Flee.

Overall, the architecture of Flee 3 symbolises a gradual evolution from a minimalistic and hard-coded heuristics-based simulation approach to a more modular and reconfigurable simulation tool. In the case of the definition of agent movement rules, we have split the functionalities across three categories: Spawning (how to create new agents), Scoring (how to calculate location and link attractiveness), and Moving (how to calculate when an agent does and does not move). Although this splitting does make it easier for developers to define new rules, many decision parameters are still defined within a static simulation settings file. In future versions of Flee, we aim to support a more powerful engine for movement rule definition. In this, we will have to first gain a better understanding of the usage patterns of Flee (which is now becoming possible due to the larger user uptake), and then trade off user convenience against development effort and deployment complexity (a consideration that is unfortunately often unaccounted for in existing domain-specific languages).

3.3. Simulation settings configuration

A main aim of Flee 3 is to support a wider range of migration scenarios. To facilitate this, we have introduced a flexible system for configuring simulation-specific parameters, replacing the limited CSV-based implementation from previous versions. Using the YAML (Yet Another Markup Language) format, Flee 3 provides a mechanism to store a broad range of parameters in a hierarchical manner. The simulation settings input file includes parameters that influence agent actions, environmental interactions, and simulation run-time. We summarise a representative range of simulation configuration settings in Table 1 and present a comprehensive list as part of the official Flee documentation [61]. Flee 3 groups simulation settings across four categories, (i) `log_levels`, which control the verbosity of the simulation in different areas, (ii) `spawn_rules`, which define how agents are spawned

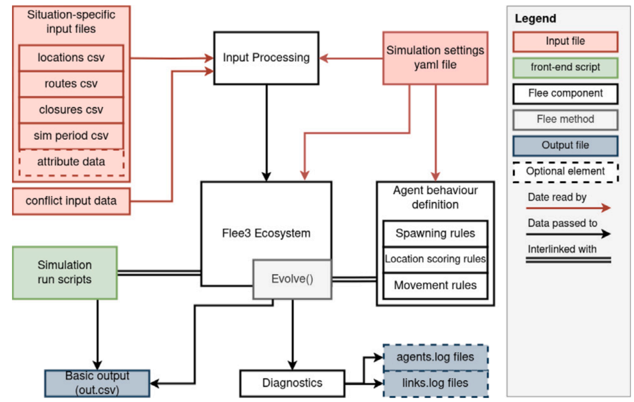


Fig. 2. Overview of the Flee 3 architecture.

in the simulation, (iii) `move_rules`, which define when and how agents move across the location graph, and (iv) `optimisations`, which contain settings that enable faster (but less accurate) simulations. These parameters collectively offer users the means to fine-tune the simulation's settings to align with their specific conflict instances and study various scenarios.

3.4. Location graph representation

Flee's simulations contain a graph-based spatial representation of geographical regions, locations, and routes that interconnect them. This spatial representation is essential, because it allows simulations users to (i) accurately track where forcibly displaced people are expected to go, (ii) introduce specific circumstances such as road closures (e.g. due to a blockade or seasonal flooding) and (iii) more easily develop a geospatial visualisation of any migration forecasts. The location graph is used to track the agents' movements within the simulation and to investigate the routes and destinations available to them, given a set level of awareness. Routes constitute the edges within this location graph representation, and could represent paved roads or paths of a different type (this aspect is discussed in depth by Boesjes et al. [63]).

Since its inception, the Flee location graph supports `conflict zone` locations, which spawn new displaced person agents, `town` locations which represent non-conflict and non-camp locations in the country of origin, and `camp` locations which represents camps, shelters and other safe havens abroad. Flee 3 extends this representation with two new core location types: the `IDPCamp` locations serve as shelters for internally displaced persons and function similarly to camps, except that they reside within the country of conflict and can be configured separately. In addition, `marker` locations indicate crossings where no settlement is present. These locations do not affect decision-making in Flee but enable post-processing tools to realistically visualise crossings in the location graphs and enable the Diagnostics module to accurately output the population breakdowns for each route. In addition, Flee 3 supports move chances and location weights that can be reconfigured based on user-defined criteria. Moreover, in Flee 3 both locations and links in the location graph can be annotated with custom attributes, allowing users to incorporate new push- and pull-factor elements such as food security and GDP. However, movement rules based on these attributes still have to be hand-coded. This is a clear limitation we hope to resolve in a future Flee version once we have a clearer understanding of the main usage patterns of this version.

In Flee, agents interact within a defined network, affecting population values by either spawning or arriving in camps. This forms the basis for simulating complex population displacement dynamics in various contexts. We present a location graph, highlighting all the main locations and route types used in Flee in Fig. 3.

Table 1
Selection of Flee simulation parameters by category. The parameter `awareness_level` was introduced in version 1.0 but was completely redefined in version 3.0.

log_levels	Description	Default	New since
`agent`	Agent verbosity level. Higher levels store all agent movements.	0	3.0
`link`	Link verbosity level. Higher levels store link populations by time step, disaggregated by demographic attributes.	0	3.0
spawn_rules	Description		
`conflict_spawn_decay`	An multiplier array describing how the number of displaced persons per day changes over time after a conflict has occurred.	gradual decline over 3 months [62].	3.0
`conflict_driven_spawning`	Base agent-spawning rates directly on conflict events.	disabled.	3.0
`camps_are_sinks`	Agents are deactivated once they reach a camp.	false.	3.0
`read_from_agents_csv_file`	Preload agents from a CSV file snapshot.	false.	3.0
move_rules	description		
`avoid_short_stints`	restrict displaced people that will take a break unless they at least travelled for a full day's distance in the last two days.	False	2.0
`awareness_Level`	Number of hops in the location graph that an agent takes into account when planning a route and detecting suitable destinations.	1 hop.	*
`capacity_scaling`	Scale-up preset camp capacities to loosen the assumptions in a validation setting.	1.0.	3.0
`camp_movechance`	Probability per day that an agent residing in a camp will move elsewhere.	0.001	1.0
`conflict_movechance`	Probability per day that an agent residing in a conflict zone will move elsewhere.	1.0	1.0
`default_movechance`	Probability per day that an agent residing in a non-camp/non-conflict zone location will move elsewhere.	0.3	1.0
`distance_power`	Adjust the importance of distance in weight calculations.	inverse linear (1.0).	3.0
`idpcamp_movechance`	Probability per day that an agent residing in an IDP camp will move elsewhere.	0.1	3.0
`max_move_speed`	Distance an agent can travel at most per day.	400km.	3.0
`softening`	adds kilometres to every link distance to reduce the importance of route length in the route planning algorithm.	10km	1.0
`use_pop_for_loc_weight`	Location population affects its attractiveness.	false.	3.1
`home_distance_power`	if positive, locations further from an agents home location are less likely to be chosen.	0.0 (disabled)	3.1
Optimisations	Description		
`hasten`	Proportionally reduce agents in simulation and validation data to reduce execution time.	1.0 (disabled)	3.0

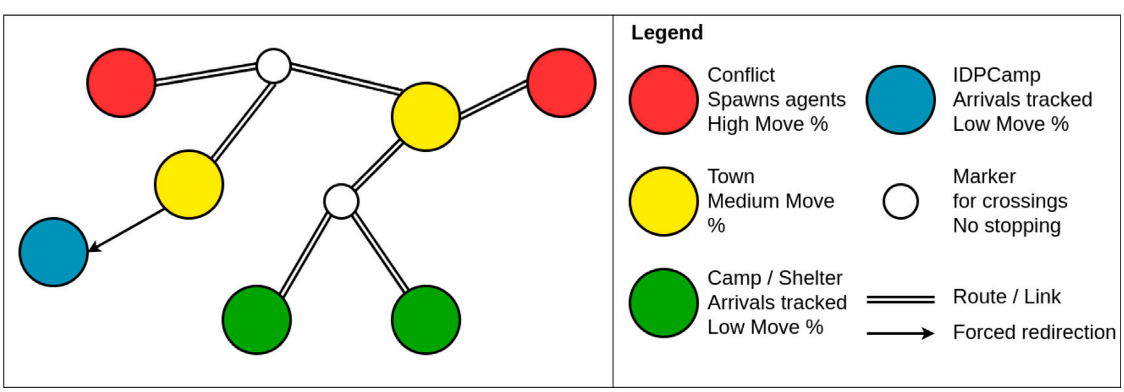


Fig. 3. Example of a location graph representation in Flee, containing all the main locations and route types. Markers and IDP Camps are new in version 3. Flee 3 features reconfigurable move chances and location weights, therefore the exact move chances depend on the user configuration.

3.5. Interaction with the location graph

Every Flee simulation operates with a dedicated Ecosystem, as illustrated in Fig. 4. This Ecosystem is the core component that encapsulates a list of all `Person` objects (or agents), the location graph, and a concept of time within the simulation, as well as managing the schedule for link closures. Technically, it is possible to employ multiple Ecosystem instances within a single Flee simulation, yet traditionally, a single instance has sufficed for our needs. In the example depicted in the figure, the Ecosystem object stores the simulation state for Flee, featuring ten agents within a fully connected location graph consisting of three locations, however, typically, location graphs in Flee are not fully connected and agents can reside on a link at the end of a time step. The persons within an Ecosystem are distributed across processes

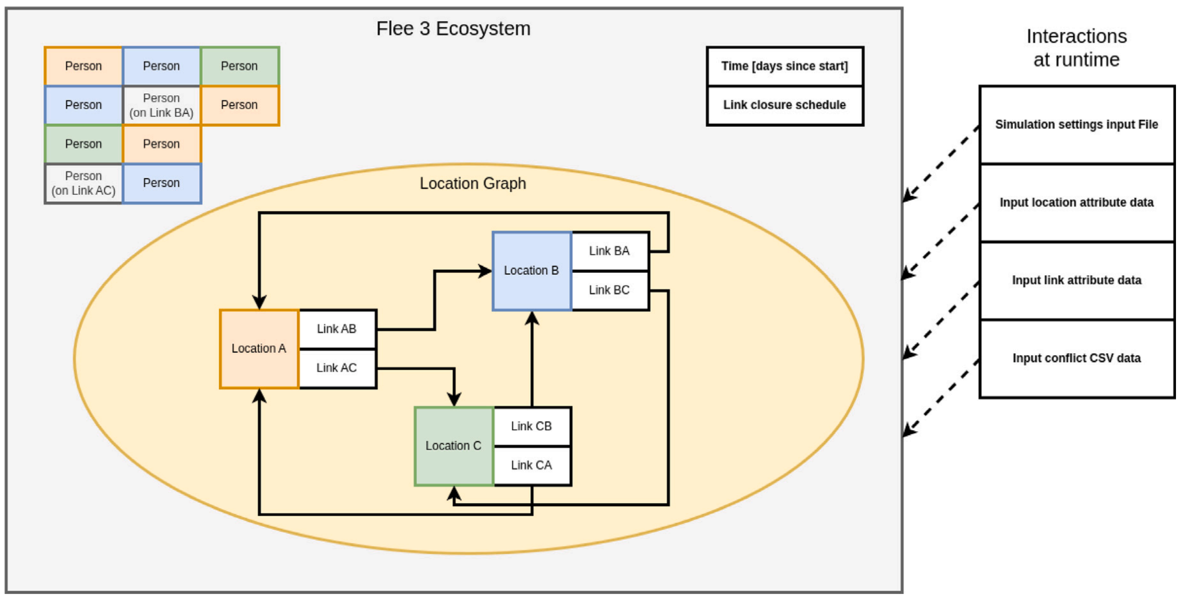


Fig. 4. Overview of the Ecosystem object, which stores the simulation state in Flee. This example features ten agents and a fully connected location graph with three locations. Agents can reside on a link at the end of a time step (or day), and location graphs are normally not fully connected.

in parallel simulations [27], while the location graph is duplicated and kept coordinated across these processes. Each person is associated with a specific location or link, indicating their current position. Furthermore, each Location in the graph is interconnected with adjacent locations through one or more unidirectional Link objects, which can be strategically used to enforce one-way forced redirections of agents. The Ecosystem is not entirely isolated; it periodically gathers data from the simulation settings input file as well as from files containing location attribute, link attribute and conflict data. The outputs are directed either to the run-script (and subsequently, recorded in the `out.csv` file) or written to separate agent and link log files. In Flee, interactions between Person agents are indirect: they primarily influence the properties of Locations and Links, which in turn affect their decision-making processes. This indirect interaction model is advantageous for parallel execution of the code. However, a limitation of this approach is that simulations requiring more frequent agent interactions than the default daily time step must be adjusted by reducing the time step size. This means shortening the intervals, such as from one day to several hours, to allow for these more frequent interactions within the same overall simulation period. This design decision reflects our prioritisation of computational efficiency and scalability in Flee's architecture.

3.6. Spawning new agents

Historically, Flee simulations are used to predict where displaced persons may go, with the total number of persons displaced over time being an underlying assumption provided in advance by the user. Essentially, for each agent spawned in a time step, the origin location is selected using a population-weighted selection algorithm, assuming that once a conflict occurs somewhere, it remains a conflict zone until the end of the simulation. In Flee 3, we eliminate this latter assumption by introducing a `conflict_decay` parameter. This parameter is configured such that older conflicts will have reduced weighting over time, with its weight reduced to a minimum of 10% of the original weight after three months. This assumption is based on a study performed in the ITFLOWS EU project [62].

One major complication of this general approach is that it can be challenging or impossible for humanitarian organisations to predict the number of expected displaced persons for conflicts that are yet to occur. To complicate things further, accurately predicting the number of displaced persons based on violence levels is a task fraught with ethical issues as well. As a result, we choose to leave precise forecasting of displacement numbers outside the main scope of Flee. However, in Flee 3, we do introduce a new mechanism to calculate the number of displacements based on basic heuristics. This feature is called `conflict-driven spawning`.

3.6.1. Conflict-driven spawning

The intention of the conflict-driven spawning feature in Flee 3 is to provide users with a straightforward mechanism to generate the number of conflict-driven displaced persons over time. When enabled, conflict-driven spawning in Flee reads data from the conflicts input file to dynamically spawn new agents. Each location is assigned a conflict intensity value for each simulated day: 0.0 indicates no conflict, 1.0 indicates full-scale violence, and, as a new feature in Flee 3, intermediate values indicate smaller-scale conflicts. In historical scenarios, Flee sources data from public databases such as ACLED [57]. However, for future conflicts, users have the flexibility to define the expected conflict evolution by integrating expert knowledge or using conflict prediction tools, such as ML models.

The process of spawning agents is directly driven by the conflicts as defined in the conflicts input file. The simulation allows for two primary mechanisms to spawn agents in response to conflict intensity. One approach involves using an absolute value; for instance, 1000 agents might be programmed to spawn in a specific location when the conflict intensity reaches a value of 1.0 on a given day. Alternatively, a fraction of the existing population can be used, such as deploying 1% of the remaining population from an area experiencing a conflict intensity of 1.0. To ensure that the model does not become too difficult to scrutinise and interpret, these spawn rates are designed to scale linearly for conflict intensities between 0.0 and 1.0.

3.7. Dynamic evolution within time steps

To provide insight into the dynamic evolution of Flee, we present a detailed overview of the tasks performed during each time steps in Flee (which equals to a single simulated day) in Fig. 5. Each time step encompasses four types of tasks: (i) all location scores are updated and any new agents are spawned in conflict zones or other sources; (ii) all agents perform an initial travel step; (iii) agents that are still moving and have no yet covered their daily distance limit will have their travel completed and (iv) the current population state for the various

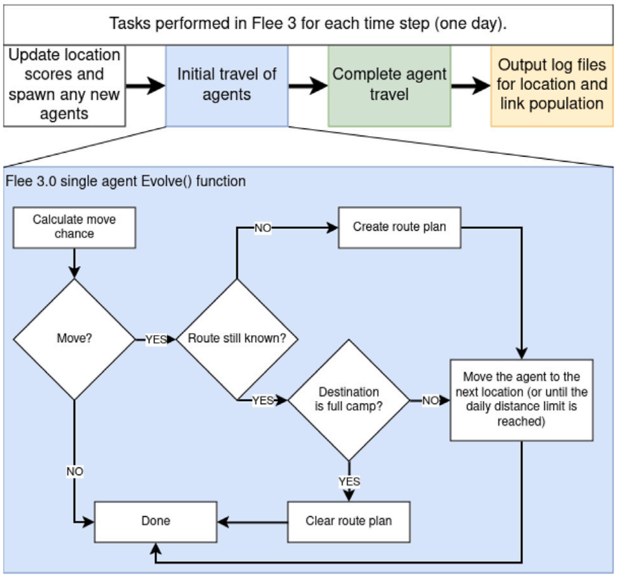


Fig. 5. High-level overview of tasks performed during a time step in Flee 3. Tasks such as move chance calculation and route selection are reconfigurable, using Flee 3 `simsetting.yml` parameters.

locations and links is written to output log files. In this design, steps (i)-(iii) are parallelised such that agents and locations are uniformly distributed across the processes, and updated in parallel (for details, please refer to Groen et al. [30]). Steps (ii) and (iii) are worth discussing in more detail, because it is in these steps that the agent movement is performed. We present the single agent decision-making algorithm for step (ii) in the bottom of Fig. 5; this algorithm is also applied for agents who are still travelling in step (iii), should they be reaching a new location.

The single agent-decision algorithm encompasses two important parts: first agents decide whether to move or not and second they decide where to travel to. Both the move chance calculation and the route selection are customisable using parameters in the simulation configuration. Where in previous versions of Flee agents had to make ad-hoc decisions about their next move during each time step, Flee 3 allows for more advanced route planning. Agents now plan their routes based on the `awareness_level` parameter, setting a maximum route length they can plan for before executing this plan. If significant changes occur at the planned destination during the journey, such as overpopulation, agents have the option to abandon their initial plan and create a new route. While on a planned route, agents can take breaks at locations, but their overall path remains preset unless a trigger condition (such as overpopulation in the planned destination) forces a route cancellation.

This new route planning approach makes the decision-making process of agents more intuitive, because it aligns more closely with real-life behaviour and is easier to compare with actual evidence. Furthermore, this approach greatly reduces computational overhead, especially when the awareness level is high, as agents do not need to recalculate their travel direction at every time step. What is still an important limitation in our algorithm is that agents will always first decide whether to move, and then choose the optimal path. This approach has benefits in terms of conceptual simplicity and ease of parallelisation. However, as a result of this design choice, the algorithm does not support a mechanism where the availability of a particularly favourable route will positively affect the likelihood that an agent wishes to move.

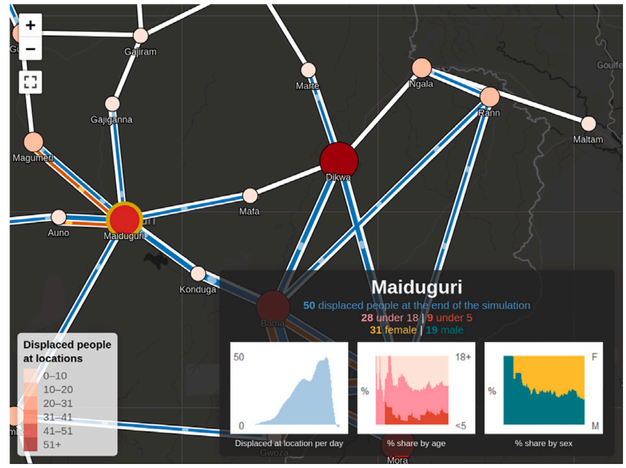


Fig. 6. Example browser-based visualisation from a Flee-based forecasting report. Details about individual locations are shown in the bottom right when a particular location is selected. Image courtesy of Save the Children.

3.8. Visualisation and interpretation

Through an array of runtime diagnostic outputs, we store updates on the behaviours of individual agents over time, their interactions, and the evolving state of the ecosystem. These diagnostic outputs can be visualised in basic graph format, or adopted by third party tools to provide more advanced and interactive visualisations. In Fig. 6 we show an example visualisation which relies on the diagnostic outputs from Flee 3. This visualisation, which is part of a preliminary visualisation approach from Save the Children, allows users to explore for example which types of agents visit which locations and which links are used most extensively during the simulation. Alternative visualisation tools for Flee have also been established over the years, for example by the World Modellers Project led by the University of Columbia [64] and the ITFLOWS project [32].

In general, visualisations using diagnostics from Flee 3 may encompass agent trajectories, population distributions, location attributes, and temporal changes. Geographic maps reveal migration routes and agent concentrations, while heat maps can be used to showcase density variations across locations. Graphs and charts present quantitative trends, aiding in the analysis of population growth, movement patterns, and environmental shifts. Having such visual outputs can support decision makers in obtaining actionable insights, identifying correlations, and uncovering causality. They may also help researchers and decision makers to reveal migration trends influenced by conflict or environmental factors, detect overcrowded regions, and explore dynamic interactions. Effective interpretation, driven by domain expertise and data analysis, bridges the gap between raw simulation data and meaningful insights, enabling decision-makers to better comprehend and respond to the complex dynamics simulated by Flee. However, a major challenge remains to establish a generic, sophisticated and open source visualisation platform for Flee: many open-source visualisation platforms lack the versatility to support the wide range of data exploration contexts needed by its users, and several past tools have become unusable over the years due to the deprecation or obsolescence of the underlying (closed-source) technological platform.

4. Experimental setup

The primary aim of Flee is to predict the arrivals of persons displaced by conflict and to do so in a timely way. To assess whether this is the case, we perform the following types of tests:

- Performance tests, to assess the time to completion and overall scalability of the code in real-world modelling situations.
- Validation tests, to assess the accuracy of Flee under different assumptions and across different conflicts.

In the case of the performance tests, we chose to run the largest real-world conflict situation that we have available: Ukraine 2022. This conflict simulation features up to 23.6 million displaced persons. For the validation tests, we chose ten other conflicts, and validated these runs against observational data from UNHCR. These runs serve to showcase how different rule sets can be effectively applied across conflicts, and how Flee 3 enables us to model an even wider range of conflict situations effectively. Note that at time of writing, we do not yet have a (UNHCR) refugee data set that is systematically stratified by one or more demographic properties. Because of these limitations, we are therefore not yet able to validate our models in a stratified manner, matching e.g. refugee arrivals by religions, gender or age.

4.1. Data integration

The effectiveness of Flee simulations relies on several external data sources. Here, we present the key steps for merging these datasets into Flee, with a focus on population and geospatial data.

To incorporate conflict events into Flee, we use ACLED, an extensive repository detailing armed conflicts and political instability. The process starts with acquiring ACLED's dataset, and consolidating conflict event details like location, date, and involved actors from various sources. We then use preprocessing tools to reformat and refine the data for simulation use. For instance, we map conflict events onto the Flee environment to capture evolving conflict zones, using modified conflict intensity values for conflict events that are relatively small or large in terms of overall magnitude. To make estimations of total displacement and for validation, we rely on data from the UNHCR archive, as discussed by Suleimenova et al. [10].

We also use population data from various sources, including census records and surveys to assign population values to individual locations in the Flee location graph. When agent properties such as gender or ethnicity are used in the simulation, we also use these data sources to create demographic distributions for each location. Each demographic attribute can be connected to Flee using a corresponding CSV file which contains one row for each possible demographic value, and one column for each location. In addition, these demographic CSV files contain a Default column, which is used if a location is not explicitly listed in one of the other columns.

Calculating accurate distances between locations is a major challenge, and there are several ways to do this. Most frequently, we rely on road-based route planning tools such as the Open Source Routing Machine (OSRM) [65]. However, in some cases, we adopt alternative approaches like walking routes or river crossings (see e.g., [12]).

4.2. Simulation workflow automation and sensitivity analysis

A key challenge in migration forecasting involves efficiently running simulations and analysing their outcomes, as illustrated in Fig. 1. To accelerate its core model, Flee 3 relies on the parallelisation strategy of its predecessor [30]. In addition we have developed the FabFlee automation tool, a plugin of the FabSim3 automation toolkit [27], to facilitate a range of automated procedures. These automated procedures are run using a single terminal command and include (i) validation of Flee 3 across ten different conflicts, (ii) sensitivity analysis using stochastic collocation to determine uncertainty caused by simulation parameters [66] (this can also be used to guide the development of new models [28]), (iii) the automated ensemble forecasting, running multiple instances to produce forecasts that account for aleatoric uncertainty or different conflict progressions [26,27]. Because Flee 3 has a much larger number of parameters than previous versions, and because parameter sensitivities can vary per strongly conflict [28] (and even more so when the code is applied to IDP problems), we intend to perform an in-depth sensitivity analysis of the code across different settings as part of a future project. One thing of interest here is to examine the relation between differentiable agent-based models and existing methods for systematic sensitivity analysis, and to explore whether automatic differentiation [67] could be of value to us to more efficiently perform sensitivity analysis for Flee 3 across large parameter spaces.

FabSim3 is also crucial for the efficient execution of Flee 3 simulations on the ARCHER2 Supercomputer and other HPC resources. Using tools such as QCG PilotJob [68] we are able to perform ensemble forecasting runs involving 1000s of Flee executions. Using this ensemble functionality, it is possible to run a batch of simulations that include a range of different conflict progressions, different interventions or even different conflicts altogether. For example, FabFlee contains automated validation command (`validate_flee`) which uses this ensemble execution functionality to compare simulation predictions across the ten different conflicts against UNHCR data, and to then calculate the validation error and generate visualisations. To help account for aleatoric uncertainty, it is possible to perform this ensemble validation with a specific number of replicated simulations. For instance, if the number of replicas is set to 20, then each conflict will be simulated 20 times, with each simulation being compared to the UNHCR validation data and the tool returning aggregate statistics of the validation results. Lastly, unless stated otherwise, all runs performed in this paper use a full single ARCHER2 node which contains 128 cores.

5. Results

5.1. Performance

Evaluating the performance of Flee 3, especially in large-scale, real-world simulations, is a crucial aspect. In many ways, Flee 3 has similar scalability characteristics to its predecessor. However, to showcase the performance of Flee in a realistic situation we present scalability results from a large and recent conflict-driven migration event (the Russian invasion of Ukraine in 2022). During this war, which started on February 28th, 2022, and we simulated for 554 days, the number of persons displaced abroad in the simulation grew from 6.34 million during the first 100 days, to about 10 million after 200 days and 23.6 million after 554 days. In production simulations, designed for real-world applications and decision-making rather than just testing and development, the number of agents typically increases as conflicts evolve, leading to more displaced persons. As a result, Flee's computational intensity and scalability grow in the simulation's later stages. This reflects the dynamics of escalating conflicts and the system's scaling behaviour.

We have conducted our simulation runs on the ARCHER2 Supercomputer, housed at the Edinburgh Parallel Computing Centre (EPCC) [69]. ARCHER2 has 5860 computing nodes, each powered by two AMD EPYC 7742 processors, cumulatively offering 128 cores per node. Furthermore, every node has 256 GB of memory. One of the key features of ARCHER2 is its advanced networking capabilities. The system is interconnected with a Hewlett Packard Enterprise (HPE) Cray Slingshot network, providing 2 × 100 Gbps of bandwidth between nodes. At the time of writing, ARCHER2 is ranked 39th in the Top 500 list of the world's most powerful supercomputers with a peak performance of approximately 20 Petaflop/s.

We provide an initial overview of the performance and scalability of Flee 3 in Fig. 7. This figure illustrates how the *wallclock* time typically decreases as we increase the number of nodes in simulations, although the logarithmic curve begins to flatten beyond 8 nodes (512 cores). In addition, we find that the simulation running for 554 days scales better than the ones running for 200 days or 100 days. This difference can be directly attributed to the problem size, as the number of displaced persons tend to increase as a conflicts evolve (and with that, the number

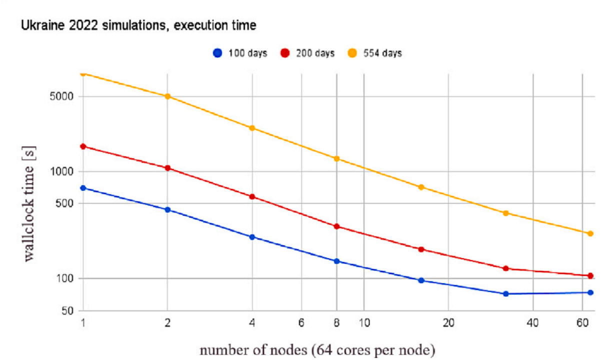


Fig. 7. Wallclock time spent simulating migration from Ukraine conflict by simulation duration, using 1 up to 64 compute nodes.

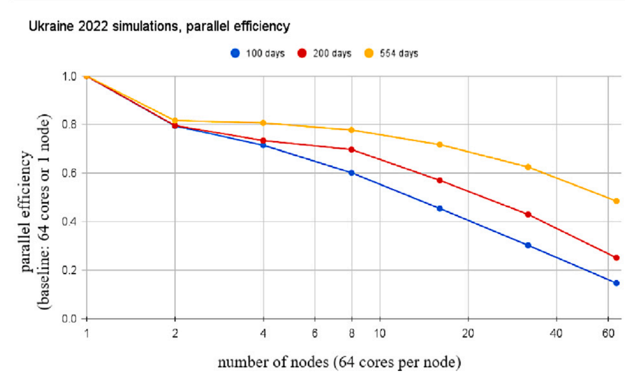


Fig. 8. Parallel efficiency of Ukraine conflict migration simulations by simulation duration, using 1 up to 64 compute nodes.

of agents in Flee simulations). Aside from constraints in the problem size, limitations in scalability are also caused by communication overheads (Flee synchronises information across locations at every time step, see Groen et al. [30]), as well as a limited divergence in agent behaviours due to the branching in our decision-making algorithm (see Fig. 5). Divergent agent behaviours can seriously impact parallel performance (see e.g. Chimeh and Richmond [70]), but in the case of Flee this impact is mitigated due to the large number of agents per core for real-life scenarios (typically more than 1000), the fact that it relies on CPUs rather than GPUs, and its non-spatial distribution of agents across its processes. Indeed, as our results show, large production simulations can be completed within less than an hour in a regime where the code is scalable. In fact, given more runtime (and cores) Flee 3 is able to support up to one billion displaced person agents like its predecessor [30], although we hope that simulations of such scale will never be required for real-world crisis response scenarios.

We present the parallel efficiency of Flee in Fig. 8, focusing on a 100-day-long simulation. In this scenario, using 64 nodes, we observe that the communication overhead begins to negatively impact the overall performance, leading to a slowdown when compared to the performance on 32 nodes. With each node on the ARCHER2 Supercomputer containing 128 cores, we find that Flee 3 runs with 60% parallel efficiency when using 512 cores for a 100 day production simulation, and slightly more efficiently for a 200 day production run. When running the conflict over its currently maximum available length (554 days), the scalability is much better, as on average the Flee simulation contains more migrating agents (i.e. the problem size becomes larger). For these runs, we still obtain a 60% parallel efficiency when using 32 nodes, or 4096 cores. Note that the runtime for our Ukraine simulation is relatively short. This is primarily because our location graph has a relatively coarse resolution and the awareness level of the agents is set to 1.

5.2. Accuracy

Measuring the accuracy of forced population displacement simulations is notoriously challenging, not least due to major challenges in collecting and curating relevant and correct validation data. In 2017, we presented a validation setting for Flee simulations which involves using aggregate UNHCR refugee data to inform how many persons are expected to be displaced, and camp-by-camp counts to validate how well our forecasts perform in different locations over time [10]. A key metric to measure the accuracy of forced population displaced forecasts in this work is the averaged relative difference (ARD), which is defined as follows:

$$E(t) = \frac{\sum_{x \in S}(|n_{sim,x,t} - n_{data,x,t}|)}{N_{data,all,t}} \quad (1)$$

Here, the predicted number of displaced persons in each camp x of the set of all camps S at time t is given by $n_{sim,x,t}$, and the measured number of displaced persons, based on UNHCR data, by $n_{data,x,t}$. The total number of displaced persons reported in the UNHCR data, aggregated across all destinations at a given time t, is given by $N_{data,all,t}$. We have slightly amended the notation of this definition from Suleimenova et al. [10], to clarify that the total number observed is given for a specific time point, and not across the whole simulation period.

As part of this paper, we perform a revised and extended validation test to assess the accuracy of Flee in historical conflict settings. In comparison to previous validation studies, we revised the assumption on camp capacities: instead of being equal to the highest value in the validation data set we now set it to 1.25 times the highest value in the validation data set. Although this new value leads to a higher averaged relative difference or ARD (+0.05 on average) relative to the UNHCR validation data, we believe it better reflects the fact that camp capacities are often not precisely known in advance.

We perform our analysis across four different rule sets to enable a side-by-side comparison: the `2.0 rule` version, which mimics the rules in Suleimenova et al. [28], but with the modified camp capacity constraint, the `modern` rule set which has a lower camp move chance, and less emphasis on distance in the link weighting (among other things), the `modern-aware` rule set which introduces a wider awareness range for each agent, and a `fastflow` rule set which has greatly increased move chances and speeds. We present the ARD relative to UNHCR observations of all validation runs in Table 2, using the rescaled method documented by Suleimenova et al. [10]. Here we find that each rule set performs differently across the validation settings, with the `fastflow` rule set obtaining the lowest ARD overall (0.377). Although the `modern` rule set obtains a slightly higher ARD on average (0.392), we do recommend this rule set for users as it contains more realistic assumptions about how fast individuals move. The `2.0 rules` version performs slightly worse still (average ARD of 0.416), although it does give the best validation score in the case of Nigeria 2016. The worst-performing rule set is `modern-aware` (average ARD of 0.446), as the enhanced awareness of agents beyond adjacent locations does not lead to an accuracy improvement in most cases. Nevertheless, this rule set does perform best in three out of ten settings, and therefore can still be useful to adopt in certain situations. When combining the best-performing runs across all four rule sets, we obtain an average ARD across all settings of 0.317, with only the South Sudan 2013 conflict resulting in an ARD above 0.5 (0.54). We also repeated the validation runs with ensemble sizes of 25 replicas per country and found that the aleatoric uncertainty in Flee runs is rather minimal, and the difference between the highest and lowest ARD in each ensemble was less than 0.01 in almost all cases (and the standard deviation less than 0.003 in all cases).

Table 2
Overview of Flee validation performance across ten different conflicts, as compared to observational data from UNHCR. We plot the averaged relative difference (ARD) of the full simulation duration for four types of algorithms, assuming camp capacity at 125% of the highest data point. These include runs with a ruleset that mimic Flee 2.0 ("2.0 rule"), as well as three Flee 3 rulesets ("modern", "modern-aware" and "fastflow") which are explained in Section 5.2. Runs with the lowest ARD for each conflict are highlighted with an asterisk.

Country	Year	2.0 rule	Average Relative Difference		
			Modern	Modern-aware	Fastflow
Nigeria	2016	0.21*	0.26	0.37	0.27
Mali	2012	0.34	0.33	0.62	0.29*
Syria	2013	0.37	0.34	0.32*	0.34
S. Sudan	2013	0.55	0.54	0.58	0.54*
Burundi	2015	0.57	0.49	0.43*	0.50
Nigeria	2022	0.47	0.47	0.51	0.45*
S. Sudan	2016	0.53	0.50	0.49	0.45*
Tigray	2020	0.40	0.38	0.36*	0.36
CAR	2013	0.41	0.39	0.44	0.35*
Mali	2017	0.33	0.22*	0.34	0.22
Average		0.416	0.392	0.446	0.377*

5.3. Relevance to practitioners

Over the years, Flee has found practical application and relevance in real-world scenarios. This subsection highlights key collaborations and tools that harness Flee's predictive power to inform decision-making and policy implementation.

One of the standout collaborations that highlight Flee's real-world impact is with ITFLOWS [32]. The ITFLOWS consortium has developed the EUMigraTool, which serves to inform NGOs and other humanitarian parties about likely migration patterns towards the EU and the current level of community tension within the EU. The tool relies on Flee 3 to provide migration forecasts in accordance with the empirical assumptions of the project and integrates these simulation results with models for large scale migration and community tension to provide a comprehensive dashboard for humanitarian partners. By including Flee-produced data, the EUMigraTool provides comprehensive insights into migration dynamics and facilitates the exchange of insights between academia, policymakers, and practitioners. A key requirement from ITFLOWS for Flee was the ability to stratify migration patterns by attributes such as gender and ethnicity, a major feature which is newly introduced in Flee 3. ITFLOWS has also been instrumental in clarifying the ethical implications of performing migration forecasts using Flee, and on the question of how long conflict events influence the rise in outbound migration.

Flee is also used in collaboration with Save the Children International [31], particularly in modelling the movement of populations displaced by conflict. In Nigeria, results of Flee have been integrated with preliminary dashboard tools from Save, and combined with emerging ML-based approaches to provide accessible periodic migration forecasts to the organisation. The feedback from Save has been instrumental in the development of Flee 3, as their operational insights led to the introduction of several new movement mechanisms in the code (such as weighting the importance of people remaining close to their home town). Currently we are investigating the application of Flee 3 in other countries (such as Mozambique) as well as for other purposes (such as modelling disaster-related migration patterns).

5.4. Ethics of migration forecasting

Forecasting migration patterns can help organisations to better understand or resolve humanitarian challenges, but there are also risks associated with it [71]. In particular, the use of forecasting results in inappropriate contexts, the over-reliance on sensitive personal data and the misuse of forecasting tools by malicious parties pose important risks. A vital consideration is that many migration forecasting models are not publicly available. As a result, they are often inaccessible for organisations that seek to provide humanitarian aid to arriving refugee, whilst organisations with fewer financial constraints (such as political organisations or governments) are able to obtain such capabilities. Because Flee is developed openly, it provides a contribution towards levelling the playing field between humanitarian support organisations and other stakeholders.

Another distinctive advantage of Flee is its basis on explicit prior assumptions without the need for training data. Flee is not a self-learning tool and therefore does not qualify as artificial intelligence. However, it is crucial to acknowledge that, like any tool, Flee's simulation outcomes can be subject to misinterpretation or configuration errors. Therefore, external scrutiny remains vital to ensure the accuracy and quality of its forecasting results. Lastly, Flee 3 does not require the use of personal information, as agents are created by sampling from demographic distribution.

6. Conclusion

We have presented Flee 3, an agent-based modelling tool designed to predict and analyse forced population displacement in conflict scenarios. Among other things, this version offers a wide range of mechanisms to define user-specific rulesets for spawning new agents and defining agent behaviour, incorporates support for demographic properties, and delivers a vastly improved algorithm for route planning. We have validated the code against UNHCR observational data from ten different conflicts. Our results have shown that the averaged relative difference between simulation forecasts and UNHCR data is 0.5 or less in all but one cases for a single unified ruleset ("fastflow"), and that the old ruleset corresponding to Flee 2.0 delivers the best validation performance for only one of the ten conflicts. We have also investigated the performance and scalability of Flee 3 in a simulation of the 2022 Ukraine conflict. Here our results show that the scalability of the code improves during runtime due to the growth of number of agents in the model, and that this large production problem can be run in full with 78% parallel efficiency using 512 cores. At time of writing we are investigating the use of grid-based location graphs, which may deliver benefits in terms of spatial accuracy but also require Flee 3 to be configured with a much higher awareness level, making runs more computationally expensive.

Through its integration with the SEAVEA toolkit,[1] users are able to apply tools such as FabSim3 and QCG-PilotJOB to efficiently run ensemble forecasts containing 1000s of simulation runs that can account for aleatoric uncertainty, as well as a diverse range of possible conflict scenarios or humanitarian interventions. The code has also been integrated with EasyVVUQ, which enables users to quantify parametric uncertainties and perform systematic sensitivity analyses. Because Flee 3 offers more flexible definition of agent rules and location/link properties, it has now become possible to investigate the use of Flee 3 in new scenarios, such as modelling internal displacement. In fact, we are exploring this new direction with the NGOs Save the Children and World Watch Research, in two separate projects, at time of writing. In addition, this latest version unifies a range of features that were presented in separate versions in previous works

Although Flee 3 delivers major improvements in terms of flexibility and scope of applicability, there are still important limitations. Most notably the code does not yet support the creation of new decision-making mechanisms by users without source code modification. This is because we do not yet have sufficient awareness of how current and future Flee users intend to customise the behavioural rules of agents. For the time being, we incorporate new mechanisms on behalf of users through GitHub issue requests, for example from Save the Children or Columbia University. Once we have a better awareness of how users typically wish to define new agent rules, we intend to develop Flee 4 and introduce a systematic way of composing custom agent behaviours (e.g., using a domain-specific language or a graphical interface).

[1] https://www.seavea-project.org

CRediT authorship contribution statement

Maziar Ghorbani: Writing – review & editing, Writing – original draft, Validation, Methodology, Data curation. **Diana Suleimenova:** Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Alireza Jahani:** Writing – review & editing, Validation, Resources, Methodology, Investigation, Data curation, Conceptualization. **Arindam Saha:** Writing – review & editing, Validation, Methodology. **Yani Xue:** Writing – review & editing, Validation, Methodology. **Kate Mintram:** Writing – review & editing, Methodology, Investigation. **Anastasia Anagnostou:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization. **Auke Tas:** Writing – review & editing, Validation, Software, Methodology, Investigation, Conceptualization. **William Low:** Writing – review & editing, Validation, Supervision, Methodology, Investigation, Conceptualization. **Simon J.E. Taylor:** Writing – review & editing, Supervision, Conceptualization. **Derek Groen:** Writing – review & editing, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Data curation, Conceptualization.

Declaration of competing interest

Nine of the authors (including myself) are Senior Lecturers, Lecturers or Associate Lecturers, and members of the Modeling & Simulation Group at Brunel University London. Our positions and affiliations with the university could be perceived as presenting a potential professional interest in the promotion and recognition of the research conducted within the university and the Modeling & Simulation Group.

Two of our co-authors are currently employed by Save the Children International. This collaboration might be perceived as a potential conflict of interest given that the research could directly benefit Save the Children International's operational strategies and decision-making processes in the field of humanitarian response.

However, we would like to affirm that our research has been conducted with the highest standards of academic integrity and objectivity. All findings and conclusions in this study are the result of rigorous scientific methodology and are not influenced by our professional affiliations or collaborations.

None of the authors have any financial interests or relationships that directly or indirectly influence the work submitted in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work is supported by the ITFLOWS, HiDALGO, and STAMINA projects, which have received funding from the European Union Horizon 2020 research and innovation program under grant agreements no 882986, 824115, and 883441. This work has also been supported by the SEAVEA ExCALIBUR project, which has received funding from EPSRC, United Kingdom under grant agreement EP/W007711/1. Simulation runs have been performed using the ARCHER2 Supercomputer, located at EPCC in Edinburgh (project e723).

References

[1] UNHCR, Figures at a glance, 2019, URL: https://www.unhcr.org/uk/about-unhcr/who-we-are/figures-glance.
[2] D. Groen, Simulating refugee movements: Where would you go? Procedia Comput. Sci. 80 (2016) 2251–2255, http://dx.doi.org/10.1016/j.procs.2016.05.400.
[3] UNHCR, Project jetson, 2023, URL: https://jetson.unhcr.org.
[4] B.Q. Huynh, S. Basu, Forecasting internally displaced population migration patterns in Syria and Yemen, Disaster Med. Public Health Prep. 14 (2020) 302–307, http://dx.doi.org/10.1017/dmp.2019.73.
[5] E. Bonabeau, Agent-based modeling: Methods and techniques for simulating human systems, Proc. Natl. Acad. Sci. 99 (2002) 7280–7287, http://dx.doi.org/10.1073/pnas.082080899.
[6] R. Brun, F. Carminati, G. Carminati, From the Web to the Grid and Beyond: Computing Paradigms Driven by High-Energy Physics, Springer Science & Business Media, 2012, http://dx.doi.org/10.1007/978-3-642-23157-5.
[7] C. Macal, Everything you need to know about agent-based modelling and simulation, J. Simul. 10 (2016) 144–156, http://dx.doi.org/10.1057/jos.2016.7.
[8] S. Taylor, Distributed simulation: State-of-the-art and potential for operational research, European J. Oper. Res. 273 (2019) 1–19, http://dx.doi.org/10.1016/j.ejor.2018.04.032.
[9] P.O. Siebers, C.M. Macal, J. Garnett, D. Buxton, M. Pidd, Discrete-event simulation is dead, long live agent-based simulation!, J. Simul. 4 (2010) 204–210, http://dx.doi.org/10.1057/jos.2010.14.
[10] D. Suleimenova, D. Bell, D. Groen, A generalized simulation development approach for predicting refugee destinations, Sci. Rep. 7 (2017) 1–9, http://dx.doi.org/10.1038/s41598-017-13828-9.
[11] C. Vanhille Campos, D. Suleimenova, D. Groen, A coupled food security and refugee movement model for the south sudan conflict, in: Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part V 19, Springer, 2019, pp. 725–732.
[12] A. Jahani, H. Arabnejad, D. Suleimanova, M. Vuckovic, I. Mahmood, D. Groen, Towards a coupled migration and weather simulation: South sudan conflict, in: M. Paszynski, D. Kranzlmüller, V.V. Krzhizhanovskaya, J.J. Dongarra, P.M. Sloot (Eds.), Computational Science – ICCS 2021, Springer International Publishing, Cham, 2021, pp. 502–515, http://dx.doi.org/10.1007/978-3-030-77977-1_40.
[13] C.J. Roy, W.L. Oberkampf, A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing, Comput. Methods Appl. Mech. Engrg. 200 (2011) 2–7, http://dx.doi.org/10.1016/j.cma.2011.03.016.
[14] S. Robinson, Conceptual modelling for simulation part ii: a framework for conceptual modelling, J. Oper. Res. Soc. 59 (2008) 291–304, http://dx.doi.org/10.1057/palgrave.jors.2602369.
[15] A. Guillén, E. Teodoro, T. Gottschalk, F. Pichierri, A. Xanthaki, K. Hansen, ITFLOWS D2.3 report on human rights, ethical, societal and data protection risks assessments, 2021, URL: https://www.itflows.eu/wp-content/uploads/2022/07/7.-D2.3-ITFLOWS-R.pdf.
[16] Y. Turgut, C.E. Bozdag, A framework proposal for machine learning-driven agent-based models through a case study analysis, Simul. Model. Pract. Theory 123 (2023) 2–6, http://dx.doi.org/10.1016/j.simpat.2022.102707.
[17] D. Helbing, S. Balietti, Social Self-Organization: Agent-Based Simulations and Experiments To Study Emergent Social Behavior, 2012, pp. 25–70, http://dx.doi.org/10.1007/978-3-642-24004-1_2, Chapter 1.
[18] F. Hosseinali, A.A. Alesheikh, F. Nourian, Agent-based modeling of urban land-use development, case study: Simulating future scenarios of Qazvin city, Cities 31 (2013) 105–113, http://dx.doi.org/10.1016/j.cities.2012.09.002.
[19] D. Karnopp, R. Rosenberg, A. Perelson, System dynamics: A unified approach, IEEE Trans. Syst. Man Cybern. 6 (1976) 724, http://dx.doi.org/10.1109/TSMC.1976.4309434.
[20] A.M. Law, W.D. Kelton, W.D. Kelton, Simulation Modeling and Analysis, vol. 3, Mcgraw-hill New York, 2007.
[21] C. Moore, C. Shalizi, Cellular Automata: A Discrete Universe, London Mathematical Society, 2016, pp. 4–16, http://dx.doi.org/10.1112/S0024609302241910.
[22] S. Wasserman, K. Faust, Social network analysis: Methods and applications, in: Structural Analysis in the Social Sciences, Cambridge University Press, 1994, http://dx.doi.org/10.1017/CBO9780511815478.
[23] S. Meyn, R. Tweedie, Markov Chains and Stochastic Stability. Vol. 92, 1993, http://dx.doi.org/10.2307/2965732.
[24] D. Fudenberg, J. Tirole, Game Theory, The MIT Press, 1991.
[25] D. Groen, D. Suleimenova, A. Jahani, Y. Xue, Facilitating simulation development for global challenge response and anticipation in a timely way, J. Comput. Sci. 72 (2023) 102107, http://dx.doi.org/10.1016/j.jocs.2023.102107.
[26] D. Suleimenova, D. Groen, How policy decisions affect refugee journeys in South Sudan: A study using automated ensemble simulations, J. Artif. Soc. Soc. Simul. 23 (2020) http://dx.doi.org/10.18564/jasss.4193.
[27] D. Groen, H. Arabnejad, D. Suleimenova, W. Edeling, E. Raffin, Y. Xue, K. Bronik, N. Monnier, P.V. Coveney, FabSim3: An automation toolkit for verified simulations using high performance computing, Comput. Phys. Comm. 283 (2023) 108596, http://dx.doi.org/10.1016/j.cpc.2022.108596.
[28] D. Suleimenova, H. Arabnejad, W.N. Edeling, D. Groen, Sensitivity-driven simulation development: a case study in forced migration, Phil. Trans. R. Soc. A 379 (2021) 2–7, http://dx.doi.org/10.1098/rsta.2020.0077.
[29] P. Anastasiadis, S. Gogolenko, N. Papadopoulou, M. Lawenda, H. Arabnejad, A. Jahani, I. Mahmood, D. Groen, P-Flee: An efficient parallel algorithm for simulating human migration, in: 2021 IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPSW, 2021, pp. 1–6, http://dx.doi.org/10.1109/IPDPSW52791.2021.00159.

[30] D. Groen, N. Papadopoulou, P. Anastasiadis, M. Lawenda, L. Szustak, S. Gogolenko, H. Arabnejad, A. Jahani, Large-scale parallelization of human migration simulation, IEEE Trans. Comput. Soc. Syst. (2023) 1–12, http://dx.doi.org/10.1109/TCSS.2023.3292932.

[31] Save the Children, Save the children international, 2023, URL: https://www.savethechildren.org.uk/.

[32] ITFLOWS Project, It tools and methods for managing migration flows, 2023, URL: https://www.itflows.eu/.

[33] I. Benenson, I. Orner, E. Hatna, Agent-Based Modeling of Householders' Migration Behavior and Its Consequences, Physica-Verlag HD, Heidelberg, 2003, pp. 97–115, http://dx.doi.org/10.1007/978-3-7908-2715-6_6, Chapter 2.

[34] J. Anderson, A. Chaturvedi, M. Cibulskis, Simulation tools for developing policies for complex systems: Modeling the health and safety of refugee communities, Health Care Manag. Sci. 10 (2007) 331–339, http://dx.doi.org/10.1007/s10729-007-9030-y.

[35] J.A. Sokolowski, C.M. Banks, R.L. Hayes, Modeling population displacement in the Syrian city of aleppo, in: Proceedings of the Winter Simulation Conference 2014, 2014, pp. 252–263, http://dx.doi.org/10.1109/WSC.2014.7019893.

[36] K. Boss, A. Groeger, T. Heidland, F. Krueger, C. Zheng, Forecasting Bilateral Refugee Flows with High-Dimensional Data and Machine Learning Techniques, Technical Report, Barcelona School of Economics, 2023, URL: https://www.itflows.eu/wp-content/uploads/2023/03/1387.pdf.

[37] C.B. Casagran, E.T. Martinez, A.G. Gil, A. Gröger, K. Boss, G. Stavropoulos, N. Gkevrekis, I. Iliopoulos, G. Papadopoulou, T. Gottschalk, et al., ITFLOWS D6.2 preliminary release of the emt, 2022, URL: https://www.itflows.eu/wp-content/uploads/2022/06/23.-D6.2-ITFLOWS.pdf.

[38] C. Searle, J. van Vuuren, Modelling forced migration: A framework for conflict-induced forced migration modelling according to an agent-based approach, Comput. Environ. Urban Syst. 85 (2021) http://dx.doi.org/10.1016/j.compenvurbsys.2020.101568.

[39] A. Termos, S. Picascia, N. Yorke-Smith, Agent-based simulation of West Asian urban dynamics: Impact of refugees, J. Artif. Soc. Soc. Simul. 24 (2) (2021) http://dx.doi.org/10.18564/jasss.4472.

[40] E. Frydenlund, P. Foytik, J.J. Padilla, A. Ouattara, Where are they headed next? modeling emergent displaced camps in the DRC using agent-based models, in: 2018 Winter Simulation Conference, WSC, 2018, pp. 22–32, http://dx.doi.org/10.1109/WSC.2018.8632555.

[41] E. Frydenlund, Modeling and simulation as a bridge to advance practical and theoretical insights about forced migration studies, J. Migr. Hum. Secur. 9 (2021) 165–181, http://dx.doi.org/10.1177/23315024211035771.

[42] R. Gore, P. Wozny, F.P.M. Dignum, F.L. Shults, C.B. van Burken, L. Royakkers, A value sensitive ABM of the refugee crisis in the netherlands, in: 2019 Spring Simulation Conference, SpringSim, 2019, pp. 1–12, http://dx.doi.org/10.23919/SpringSim.2019.8732867.

[43] M. Simon, C. Schwartz, D. Hudson, S. Johnson, A data-driven computational model on the effects of immigration policies, Proc. Natl. Acad. Sci. 115 (2018) 2–10, http://dx.doi.org/10.1073/pnas.1800373115.

[44] C. Boshuijzen-van Burken, R. Gore, F. Dignum, L. Royakkers, P. Wozny, F.L. Shults, Agent-based modelling of values: The case of value sensitive design for refugee logistics, J. Artif. Soc. Soc. Simul. 23 (6) (2020) http://dx.doi.org/10.18564/jasss.4411.

[45] M. Carammia, S.M. Iacus, T. Wilkin, Forecasting asylum-related migration flows with machine learning and data at scale, Sci. Rep. 12 (2022) 1–10, http://dx.doi.org/10.1038/s41598-022-05241-8.

[46] L. Estrada, D. Groen, J. Ramirez-Marquez, A serious video game to support decision making on refugee aid deployment policy, Procedia Comput. Sci. 108 (2017) 205–214, http://dx.doi.org/10.1016/j.procs.2017.05.112.

[47] Y. Xue, M. Li, H. Arabnejad, D. Suleimenova, A. Jahani, B.C. Geiger, Z. Wang, X. Liu, D. Groen, Camp location selection in humanitarian logistics: A multiobjective simulation optimization approach, in: Computational Science - ICCS 2022, Springer International Publishing, Cham, 2022, pp. 1–8, http://dx.doi.org/10.1007/978-3-031-08757-8_42.

[48] O.B. Bayraktar, D. Günneç, F.S. Salman, E. Yücel, Relief aid provision to en route refugees: Multi-period mobile facility location with mobile demand, European J. Oper. Res. 301 (2022) 2–9, http://dx.doi.org/10.1016/j.ejor.2021.11.011.

[49] A. Klabunde, F. Willekens, Decision-making in agent-based models of migration: State of the art and challenges, Eur. J. Popul. 32 (2016) 1–5, http://dx.doi.org/10.1007/s10680-015-9362-0.

[50] J. Thober, N. Schwarz, K. Hermans, Agent-based modeling of environment-migration linkages: A review, Ecol. Soc. 23 (2018) 1–4, http://dx.doi.org/10.5751/ES-10200-230241.

[51] A. McAlpine, L. Kiss, C. Zimmerman, Z. Chalabi, Agent-based modeling for migration and modern slavery research: a systematic review, J. Comput. Soc. Sci. 4 (2021) 1–7, http://dx.doi.org/10.1007/s42001-020-00076-7.

[52] G. De Luca, T.J. Lampoltshammer, S. Parven, J. Scholz, A literature review on the usage of agent-based modelling to study policies for managing international migration, Soc. Sci. 11 (2022) http://dx.doi.org/10.3390/socsci11080356.

[53] G.J. Abel, M. Brottrager, J. Crespo Cuaresma, R. Muttarak, Climate, conflict and forced migration, Global Environ. Change 54 (2019) 2–7, http://dx.doi.org/10.1016/j.gloenvcha.2018.12.003.

[54] K.H. Pham, M. Luengo-Oroz, Predictive modelling of movements of refugees and internally displaced people: towards a computational framework, J. Ethn. Migr. Stud. 49 (2023) 1–9, http://dx.doi.org/10.1080/1369183X.2022.2100546.

[55] A.C. Adib Bencherif, D. Stockemer, Lethal violence in civil war: Trends and micro-dynamics of violence in the Northern Mali conflict (2012–2015), Stud. Confl. Terror. 46 (2023) 1–4, http://dx.doi.org/10.1080/1057610X.2020.1780028.

[56] D. Groen, D. Bell, H. Arabnejad, D. Suleimenova, S.J. Taylor, A. Anagnostou, Towards modelling the effect of evolving violence on forced migration, in: 2019 Winter Simulation Conference, WSC, IEEE, 2019, pp. 297–307.

[57] C. Raleigh, r. Linke, H. Hegre, J. Karlsen, Introducing ACLED: An armed conflict location and event dataset, J. Peace Res. 47 (2010) 651–660, http://dx.doi.org/10.1177/0022343310378914.

[58] S. Davies, T. Pettersson, M. Öberg, Organized violence 1989–2021 and drone warfare, J. Peace Res. 59 (2022) 593–610.

[59] D.W. Wright, R.A. Richardson, W. Edeling, J. Lakhlili, R.C. Sinclair, V. Jancauskas, D. Suleimenova, B. Bosak, M. Kulczewski, T. Piontek, et al., Building confidence in simulation: applications of easyvvuq, Adv. Theory Simul. 3 (2020) 1900246.

[60] A. Chopra, R. Raskar, J. Subramanian, B. Krishnamurthy, E.S. Gel, S. Romero-Brufau, K.S. Pasupathy, T.C. Kingsley, Deepabm: scalable and efficient agent-based simulations via geometric learning frameworks-a case study for COVID-19 spread and interventions, in: 2021 Winter Simulation Conference, WSC, IEEE, 2021, pp. 1–12.

[61] Flee Documentation, Flee simulation settings (simsetting.yml), 2023, URL: https://flee.readthedocs.io/en/master/Simulation_settings_basic/.

[62] M. Di Salvo, A. Ounnas, D. Postica, ITFLOWS D3.4 time sequence of forced displacement into neighbouring countries, 2021, URL: https://www.itflows.eu/wp-content/uploads/2022/06/25.-D3.4-ITFLOWS.pdf.

[63] F. Boesjes, A. Jahani, B. Ooink, D. Groen, Analysing the effect of a dynamic physical environment network on the travel dynamics of forcibly displaced persons in Mali, Int. J. Netw. Dyn. Intell. (2024) 100003.

[64] M. Puma, World modellers project, 2022, URL: https://people.climate.columbia.edu/projects/view/57.

[65] S. Huber, C. Rust, Calculate travel time and distance with openstreetmap data using the open source routing machine (OSRM), Stata J. 16 (2016) 2–5, http://dx.doi.org/10.1177/1536867X1601600209.

[66] D. Suleimenova, H. Arabnejad, W.N. Edeling, D. Coster, O.O. Luk, J. Lakhlili, V. Jancauskas, M. Kulczewski, L. Veen, D. Ye, et al., Tutorial applications for verification, validation and uncertainty quantification using VECMA toolkit, J. Comput. Sci. 53 (2021) 101402.

[67] P. Andelfinger, Towards differentiable agent-based simulation, ACM Trans. Model. Comput. Simul. 32 (2023) 1–26.

[68] B. Bosak, T. Piontek, P. Karlshoefer, E. Raffin, J. Lakhlili, P. Kopta, Verification, validation and uncertainty quantification of large-scale applications with qcg-pilotjob, in: International Conference on Computational Science, Springer, 2021, pp. 495–501.

[69] EPCC, Epcc for high-performance computing services, 2023, URL: https://www.epcc.ed.ac.uk/high-performance-computing-services.

[70] M.K. Chimeh, P. Richmond, Simulating heterogeneous behaviours in complex systems on gpus, Simul. Model. Pract. Theory 83 (2018) 3–17.

[71] ITFLOWS Consortium, Policy brief no. 5, 2023, URL: https://www.itflows.eu/wp-content/uploads/2022/06/ITFLOWS-Policy-Brief-5-D8.1.pdf.

Dr. Maziar Ghorbani is an Associate Lecturer in Computer Science with a Ph.D. in Experimental Physics. His research spans mathematics, electronics engineering, Computer Science, and experimental physics. He has contributed to various European and international projects, including the High-Luminosity Upgrade of the Large Hadron Collider, Horizon, STAMINA, SEAVEA, and ITFLOWS, focusing on large-scale simulations and predictive models. His focus is on High performance Distributed Computing and automation framework for Run-time Distributed Architecture for large-scale simulations.

Dr. Diana Suleimenova is a Lecturer in the Department of Computer Science at Brunel University London. Her research concentrates on agent-based modelling, forced displacement prediction, and verification, validation and uncertainty quantification (VVUQ) of multiscale applications deployed on emerging exascale platforms. Her research led to a range of international collaborations on developing and analysing the sensitivities of new migration models. She is co-Chair of the Multiscale Modelling and Simulation workshop and a Knowledge Exchange coordinator for the Software Environment for Actionable and VVUQ-evaluated Applications (SEAVEA), which aims to develop an exascale-ready toolkit for VVUQ techniques in application to various domains.

Dr. Alireza Jahani specialises in Agent-Based Modelling, Multi-Agent Systems, Machine Learning, and Knowledge Management. Before joining Brunel University London, he was the Deputy of Technical and Information Services and Assistant Professor at Faculty of Information Technology, Mehralborz University (MAU), Iran. He obtained his Ph.D. in Computer Science from University Putra Malaysia (UPM) in 2014 and he received his Master of Science in e-Commerce from Iran University of Science and Technology in 2008. Alireza is the main developer of DFlee, which is an extension of the Flee code to facilitate the modelling of disaster evacuation.

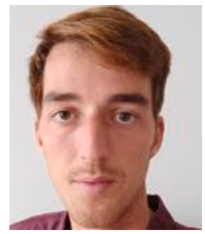
Dr. Arindam Saha is a researcher specialising in dynamical systems and complex networks. With a background in extreme events and delay-coupled relaxation oscillators, he has expanded his expertise to agent-based disease spread modelling, focusing on verifying, calibrating, and validating computational models. He has held research positions at prestigious institutions like Brunel University London and the University of Sheffield, working on projects related to exascale applications, pandemic prediction, and swarm awareness. He has published extensively in reputable journals, demonstrating his contributions to topics such as multi-agent systems, oscillator synchronisation, extreme events, and piece-wise smooth maps.

Dr. Yani Xue is a Lecturer at Brunel University London. She received a Ph.D. degree in Computer Science from Brunel University London in 2021, and currently works on many-objective optimisation in the context of migration modelling through the EU-funded ITFLOWS project. Her main research interests include multi/many-objective optimisation, evolutionary computation, search-based software engineering, engineering applications, simulation, and high performance data analytics.

Dr. Kate Mintram is an Associate Lecturer in the Department of Computer Science at Brunel University London. Prior to this, she was a Research Fellow working in the STAMINA project, a Horizon2020 funded project which aims to develop a toolset for pandemic crisis prediction and management within and across European borders.

Kate's background is in Environmental Biology . She completed her B.Sc. at the University of Plymouth in 2014 with an industrial placement year at AstraZeneca, and completed her Ph.D. in agent-based simulation at the University of Exeter in 2019 within the College of Life and Environmental Sciences. Her Ph.D. and subsequent postdoctoral research centred around the development and application of agent-based models for realistic predictions of chemical exposure effects on fish populations. Prior to joining Brunel, Kate also spent a year as an animal welfare consultant in the NGO sector, and is an Associate Fellow of the Higher Education Academy.

Dr. Anastasia Anagnostou is a Senior Lecturer in Computer Science at the College of Engineering, Design and Physical Sciences, Department of Computer Science, Brunel University London, UK. She holds a Ph.D. in Hybrid Distributed Simulation, an M.Sc. in Telemedicine and e-Health Systems and a B.Sc. in Electronics Engineering. Her research interests are related to the application of modelling and simulation techniques in the Healthcare and Industry.



Auke Tas is a Data Scientist at Save the Children International. He has a M.Sc. in Econometrics from the Erasmus University in Rotterdam and expertise in quantitative analysis ranging from the mathematical fundamentals, choosing and programming a model and translating quantitative inferences to a real-world narrative. Auke has experience with a wide range of models, both supervised and unsupervised using all sorts of data sources both structured and unstructured.

William Low is a Programme Lead at the Digital Solutions Migration and Displacement Initiative at Save the Children International. He has more than 15 years of experience in the international development and humanitarian sector. Initially, he was in management and operational positions, later in data, GIS and innovation roles focused on migration and displacement, climate and urban. He has a considerable experience in the Asia and Pacific region, primarily Myanmar.

Prof. Dr. Simon J.E. Taylor is a Professor of Computing and the director of the Modelling & Simulation Group in the Department of Computer Science, Brunel University London (https://tinyurl.com/ya5zjh8z). He has a successful track record of research in Modelling & Simulation, particularly with high performance distributed simulation in industry through digital infrastructures. He has developed these solutions with over 20 companies resulting in over 5M GBP in cost savings and increased production. He has successfully applied these experiences in the development of digital infrastructures in Africa and has led innovation in over 10 African countries. He has also led the development of international standards in these areas. He regularly audits and chairs commercial project reviews in distributed computing and Modelling & Simulation. He a member of the ACM SIGSIM Steering Committee and founder of the Journal of Simulation. He has chaired several major conferences and his published over 150 articles.

Dr. Derek Groen is a Reader in Computer Science at Brunel University London, and a Visiting Lecturer at University College London. He has a Ph.D. from the University of Amsterdam (2010) in Computational Astrophysics, and was a Post-Doctoral Researcher at UCL for five years prior to joining Brunel as Lecturer. Derek has a strong interest in high performance simulations, multiscale modelling and simulation, and so-called VVUQ (verification, validation and uncertainty quantification). In terms of applications he is a lead developer on the Flee migration modelling code and the Flu And Coronavirus Simulator (FACS) COVID-19 model. He has also previously worked on applications in astrophysics, materials and bloodflow. Derek has been PI for Brunel in two large and recent Horizon 2020 research projects (VECMA on uncertainty quantification, and HiDALGO on global challenge simulations) and he is currently the technical manager of the UK-funded SEAVEA project which develops a VVUQ toolkit for large-scale computing applications (seaveaproject.org).