Prime Locations: Agent-based Model Appendix

Gabriel M. Ahlfeldt* Thilo N. H. Albers† Kristian Behrens‡

September 22, 2020

Abstract

This appendix provides information on the agent-based model that we use in Ahlfeldt, Albers, and Behrens (2020) to generate the simulation output underlying Tables 3 and 4, as well as the simulation output used in Appendix A.5. The complete C++ source code of the model will be made available online at a later stage of the project.

*London School of Economics and Political Sciences (LSE), CEPR, CESifo, and CEP; g.ahlfeldt@lse.ac.uk.

†Humboldt University Berlin; thilo.nils.hendrik.albers@hu-berlin.de.

‡Université du Québec à Montréal (UQAM), National Research University Higher School of Economics, and CEPR; behrens.kristian@uqam.ca.

1 Agent-based model

This appendix provides details on the agent-based model (ABM). We successively describe: workers; firms and worker-firm matching; the construction sector; firm movement; equilibrium; endogenous network formation; shocks; and the parametrization of the model. We also provide succinct details on the technical implementation. The complete C++ source code of the model will be made available online at a later stage of the project.

1.1 Workers

There is a fixed set of workers $\omega \in \Omega$. Each worker has Cobb-Douglas preferences over two goods, housing h and traded consumption goods, g. Housing costs r_ℓ per unit in location ℓ, whereas the price of the traded good is the same everywhere and normalized to one. Conditional on living in ℓ, consumer ω solves the following optimization problem:

$$\max_{h,g} \;\; U_\omega(h, g \mid \ell) = \frac{A_{\ell\omega}}{\alpha^\alpha (1-\alpha)^{1-\alpha}} h^\alpha g^{1-\alpha} \quad \text{s.t.} \quad r_\ell h + g = w_\omega^{\text{net}}, \tag{1}$$

where w_ω^{net} denotes the consumer's net income (net wage); and where $A_{\ell\omega}$ denotes a (possibly consumer-specific) preference for location ℓ. In what follows, we abstract from the latter by letting $A_{\ell\omega} \equiv 1$. The solution to the consumer's problem (1) is such that

$$h^* = \frac{\alpha w_\omega^{\text{net}}}{r_\ell} \equiv h_{\ell\omega} \quad \text{and} \quad g = (1-\alpha) w_\omega^{\text{net}} \equiv g_\omega. \tag{2}$$

Indirect utility in ℓ is then given by

$$U_\omega^* \,\big|_\ell = A_{\ell\omega} w_\omega^{\text{net}} r_\ell^{-\alpha} \tag{3}$$

and aggregate demand for housing by workers in location ℓ by

$$H_\ell^d(\text{worker}) = \sum_{\omega \in \ell} h_{\ell\omega} = \frac{\alpha}{r_\ell} \sum_{\omega \in \ell} w_\omega^{\text{net}}.$$

1.2 Firms and worker-firm matching

Firm φ matches with a *set of* workers to produce a nationally traded prime service. We assume the price of prime services is determined outside the city and normalized to one. Technology is such that a firm-worker match in location ℓ produces output of value

$$\mu_{\omega\varphi\ell} = B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi, \tag{4}$$

where $\theta_\omega > 0$ and $\theta_\varphi > 0$ are the worker's and the firm's productivity parameters; and where $B_{\ell\varphi} = f(b_\ell, Y_{\ell\varphi})$ denotes a firm-specific productivity shifter in location ℓ. We assume the latter

depends on a location-specific underlying fundamental productivity, B_ℓ, and a firm-location-specific production externality, $Y_{\ell\varphi}$. Letting $E_{\ell\setminus\varphi}$ and E_j denote the number of employees working in other firms located in ℓ and in some other location j, we assume

$$B_{\ell\varphi} = B_\ell \times Y_{\ell\varphi}, \quad \text{with} \quad Y_{\ell\varphi} = \frac{1}{2} \times \left[\left(1 + E_{\ell\setminus\varphi}\right)^{\epsilon} + \left(1 + \sum_{j \in \mathcal{N}(\ell)} E_j\right)^{\epsilon_n} \right], \tag{5}$$

where $\mathcal{N}(\ell)$ is the set of neighbors of location ℓ over which external effects operate; and $0 \leq \epsilon_n \leq \epsilon$ is the strength of the production externality, broken down into location- and neighborhood-specific parts. Firms take $B_{\ell\varphi}$ as given. Note that expressions (4) and (5) imply that: (i) more productive workers match on average with more productive firms; and (ii) more productive firms benefit more from agglomeration externalities.

We assume that output is split between workers and firms with a constant share ν going to the firm φ (this may be viewed as the outcome of Nash bargaining).[1] Workers are paid according to their rank in the firm. In particular, the technology is such that workers higher up in the hierarchy (with lower rank in the firm) generate more gross output for the firm because they contribute less to SOC costs. To deal with this, we keep track of workers' ranks in the firms to determine their wages. All workers with rank $R_\omega \leq \overline{R}_\varphi$ generate value

$$B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi - \xi r_\ell \tag{6}$$

for the firm, of which a share $1 - \nu$ is paid as wages:

$$w_\omega|_\varphi = (1-\nu)\big(B_{\ell\varphi}\theta_\omega\theta_\varphi - \xi_\varphi r_\ell\big), \quad R_\omega \leq \overline{R}_\varphi. \tag{7}$$

If a worker is hired at a rank above the SOC threshold $\overline{R}_\varphi$, he generates less surplus and receives, therefore, lower wage:

$$w_\omega|_\varphi = (1-\nu)\Big[B_{\ell\varphi}\theta_\omega\theta_\varphi - \xi r_\ell - \left(R_\omega^{\epsilon_\varphi} - (R_\omega - 1)^{\epsilon_\varphi}\right)\Big], \quad R_\omega > \overline{R}_\varphi. \tag{8}$$

Summing expressions (7) and (8) across all workers in the firm, we have (suppressing the constant factor $1 - \nu$ in what follows):

$$\begin{aligned}
\pi_\varphi^{\text{gross}} &= \sum_{R_\omega \leq \overline{R}_\varphi} \left[B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi - \xi r_\ell\right] + \sum_{R_\omega > \overline{R}_\varphi} \left[B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi - \xi r_\ell - \left(R_\omega^\epsilon - (R_\omega - 1)^\epsilon\right)\right] \\
&= \sum_{\omega \in N_\varphi} \left[B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi - \xi r_\ell\right] - \sum_{R_\omega > \overline{R}_\varphi} \left[R_\omega^\epsilon - (R_\omega - 1)^\epsilon\right] \\
&= \sum_{\omega \in N_\varphi} \left[B_{\ell\varphi} \times \theta_\omega \times \theta_\varphi - \xi r_\ell\right] - \left(R_{\max}^\epsilon - \overline{R}_\varphi^\epsilon\right) = B_{\ell\varphi} \times \theta_\varphi \sum_{\omega \in N_\varphi} \theta_\omega - |N_\varphi| \xi r_\ell - \left(|N_\varphi|^\epsilon - \overline{R}_\varphi^\epsilon\right),
\end{aligned}$$

[1] We can easily allow for firm-specific ν_φ parameters, but there is little to be gained from doing so. In Ahlfeldt et al. (2020) we hence assume the split is independent of the firm.

where $R_{\max} = |N_\varphi|$ is the maximum rank of a worker in the firm (which is also equal to the size of the firm in terms of workers). Hence, the gross profit, the net profit, and the wage bill of a firm are given by:

$$\begin{aligned} \pi_\varphi^{\text{gross}} &= B_{\ell\varphi}\theta_\varphi \sum_{\omega\in N_\varphi} \theta_\omega - |N_\varphi|\xi r_\ell - \left(|N_\varphi|^\epsilon - \overline{R}_\varphi^\epsilon\right), \quad \pi_\varphi^{\text{net}} = \nu\pi_\varphi^{\text{gross}} \\ \text{wagebill}_\varphi &= (1-\nu)\pi_\varphi^{\text{gross}} = \sum_{\omega\in N_\varphi} w_\omega|_\varphi \end{aligned}$$

where individual wages are given by (7) and (8).

Observe that workers' and firms' incentives are aligned since they split the surplus generated by the match. Hence, it is rational for a firm to hire workers that produce positive match values and to reject matches that produce negative match values. We assume that firms accept all workers who add a positive amount of profit and reject workers who do not add positive profit. Technically, workers will not accept a job that does not pay positive wage (since their outside option is zero), and since wage is a share of profit it is enough to check on the workers' side for the acceptability of the matches. There are of course a large number of possible matches between firms and workers. The exact matching pattern that is realized depends, in particular, on the order in which workers search across firms. To reduce the number of matching patterns, we focus on 'efficient patterns': positive assortative ones. To this end, we start with an empty assignment and proceed sequentially in decreasing order of workers' and firms' productivities. This produces matches with high aggregate match values that are 'efficient'. Of course, we have a problem of many-to-one matching with decreasing returns and differences in costs (via the rents, which are endogenous and depend on the firm's location). Hence, the problem is significantly more complicated than a 'standard supermodular assignment problem'.

We define a *stable assignment without side payments* as an assignment where: (i) all worker-firm matches produce positive value; and (ii) no worker can find a better match than the one he is currently in. Note that we do not place any re-optimization conditions on the firms for excluding side payments (i.e., firms cannot poach workers from other firms by proposing side payments that split the additional surplus generated by such switches). There are also two other potential types of re-optimization.

Re-optimization within firms. This type of reoptimization is never profitable to workers or firms, irrespective of the existence of side payments. Consider two workers ω and ω' in firm φ, with ranks $R_\omega < R_{\omega'}$. Switching the ranks of the workers leaves the total surplus generated for the firm unchanged, i.e., there are no incentives for the firm to switch workers' positions. On the worker side, the match value is independent of the rank, whereas higher rank implies non-increasing wage (constant match value but higher SOC costs). Hence, if $\omega \neq \omega'$, at least one of the worker looses from the switch. Since the sum of workers' wages remains unchanged, there is no possible side payment to make this change mutually beneficial to the workers.

Re-optimization between firms. Consider two workers ω and ω' in firms φ and φ', with ranks $R_{\omega,\varphi} < R_{\omega',\varphi'}$. Assuming no side payments, the profit of firm φ changes by $\Delta\pi_\varphi = (\theta_{\omega'} - \theta_\omega)\theta_\varphi$, whereas the profit of firm φ' changes by $\Delta\pi_{\varphi'} = (\theta_\omega - \theta_{\omega'})\theta_{\varphi'}$ (SOC costs and costs for office space are unchanged). Hence, the only mutually acceptable switch where $\Delta\pi_{\varphi'} \geq 0$ and $\Delta\pi_\varphi \geq 0$, is such that $\theta_\omega = \theta_{\omega'}$, which will never occur since there is nothing to be gained for the firm (recall that workers bear the commuting costs, and that the firms' size remains unchanged by the switch, i.e., it pays the same SOC cost and total land rent).

In the implementation, we hence assume that workers search optimally across firms, starting with the most productive workers and the most productive firms. This gives us an initial allocation. Then, during each iteration workers may search across firms to determine whether they can find a better match as firms move and land rents adjust in the different locations.

1.3 Construction sector

The construction sector combines land, L, and capital, K, to produce real estate that can be used for either housing or offices. We consider, for simplicity, the case of perfect substitutes between residential housing and commercial real estate. The construction sector is also perfectly competitive. Assume that the supply of constructible land in each location ℓ is given by $\overline{L}_\ell$. The production function for housing is given by

$$H_\ell = \frac{D_\ell}{\beta^\beta(1-\beta)^{1-\beta}} L_\ell^\beta K_\ell^{1-\beta}, \tag{9}$$

where D_ℓ is a location-specific productivity shifter related to the fundamental conditions in that location. Housing is sold at a rental price r_ℓ per unit to both residents and firms. Normalizing the price of perfectly mobile capital to one and letting ρ_ℓ stand for the price of land in ℓ, profit maximization implies

$$\max_{L_\ell, K_\ell} \quad \pi_\ell = r_\ell \frac{D_\ell}{\beta^\beta(1-\beta)^{1-\beta}} L_\ell^\beta K_\ell^{1-\beta} - \rho_\ell L_\ell - K_\ell \tag{10}$$

The first-order conditions of (10) are given by:

$$\begin{aligned} \beta r_\ell \frac{D_\ell}{\beta^\beta(1-\beta)^{1-\beta}} L_\ell^{\beta-1} K_\ell^{1-\beta} &= \rho_\ell \\ (1-\beta) r_\ell \frac{D_\ell}{\beta^\beta(1-\beta)^{1-\beta}} L_\ell^\beta K_\ell^{-\beta} &= 1, \end{aligned}$$

which directly implies that $\frac{K_\ell}{L_\ell} = \rho_\ell \frac{1-\beta}{\beta} \;\Rightarrow\; L_\ell = \frac{1}{\rho_\ell}\frac{\beta}{1-\beta} K_\ell$. This allows us to derive the unit factor demands from the following condition:

$$\frac{D_\ell}{\beta^\beta(1-\beta)^{1-\beta}} \left[\frac{1}{\rho_\ell}\frac{\beta}{1-\beta} K_\ell\right]^\beta K_\ell^{1-\beta} = 1,$$

which yields

$$K_\ell^{\text{unit}} = \frac{1-\beta}{D_\ell}\rho_\ell^\beta \quad \text{and} \quad L_\ell^{\text{unit}} = \frac{\beta}{D_\ell}\rho_\ell^{\beta-1}.$$

Land supply equals land demand in equilibrium. Assume that $z_\ell \overline{L}_\ell$ is the supply, where $\overline{L}_\ell$ is the available surface and $z_\ell \in [0,1]$ the developable share (which captures, e.g., zoning restrictions). We then have

$$H^d(r_\ell) L_\ell^{\text{unit}} = H^d(r_\ell)\frac{\beta}{D_\ell}\rho_\ell^{\beta-1} = z_\ell \overline{L}_\ell,$$

which yields the price of land

$$\rho_\ell^{\beta-1} = \frac{1}{H^d(r_\ell)}\overline{L}_\ell \frac{z_\ell D_\ell}{\beta}. \tag{11}$$

Unit costs for building housing are then given by

$$c(\rho_\ell) = \rho_\ell L_\ell^{\text{unit}} + K_\ell^{\text{unit}} = \rho_\ell \frac{\beta}{D_\ell}\rho_\ell^{\beta-1} + \frac{1-\beta}{D_\ell}\rho_\ell^\beta = \frac{\rho_\ell^\beta}{D_\ell}$$

and this must equal the unit price of housing, r_ℓ, because of zero profits in the housing sector:

$$r_\ell = c(\rho_\ell) \quad \Rightarrow \quad r_\ell = \frac{\rho_\ell^\beta}{D_\ell} \quad \Rightarrow \quad \rho_\ell = (D_\ell r_\ell)^{1/\beta} \tag{12}$$

Combining (11) and (12), we thus need to solve

$$r_\ell^{\frac{\beta-1}{\beta}} = \frac{1}{H^d(r_\ell)}\overline{L}_\ell \frac{z_\ell D_\ell^{\frac{1}{\beta}}}{\beta} \quad \Rightarrow \quad H^d(r_\ell) = \overline{L}_\ell \frac{z_\ell D_\ell^{\frac{1}{\beta}}}{\beta} r_\ell^{\frac{1-\beta}{\beta}}$$

for the rent r_ℓ, where the aggregate housing demand in location ℓ is given by

$$H^d(r_\ell) = \sum_{\omega \in \ell} \frac{\alpha_\omega w_\omega^{\text{net}}}{r_\ell} + \sum_{\varphi \in \ell} \xi |N_\varphi|. \tag{13}$$

Since we have redundant parameters that have no separate effect, we can use the following normalization (for a 'composite productivity parameter') $\widetilde{D_\ell} \equiv z_\ell D_\ell^{1/\beta}/\beta > 0$, so that the housing supply curve is given by $\overline{L}_\ell \widetilde{D_\ell} r_\ell^{(1-\beta)/\beta}$.

1.4 Firm movement

Firms are free to move between locations. We assume that firms move between locations by (temporarily) maintaining their employment relationships. Stated differently, a firm moves between locations keeping its current workforce; and workers in the firm keep their current rank. A firm searches over all locations to find the location that maximizes its profit conditional on its current set $|N_\varphi|$ or employees and the vector of land rents $\mathbf{r}$. Firm's profit is given by

$$\pi_\varphi|_\ell = B_{\ell\varphi}\theta_\varphi \sum_{\omega \in N_\varphi} \theta_\omega - \text{SOC}(|N_\varphi|) - \xi |N_\varphi| r_\ell.$$

The elements that change with location ℓ are $B_{\ell\varphi}$ (agglomeration externalities), and $\xi|N_\varphi|r_\ell$ (costs of office space); the aggregate matching value $\theta_\varphi \sum_{\omega\in N_\varphi}\theta_\omega$ and the SOC costs $\text{SOC}(|N_\varphi|)$ are independent of the location conditional on the set of employees N_φ. Of course, once firms move these latter two terms can also change since: (i) rents will changes; and (ii) workers can rematch by finding better employers (recall that they pay commuting costs and that changes in land rent also change the value they create for the firm).

We assume that there can be small but positive moving costs $\eta_\varphi \geq 0$. Formally, a firm moves if and only if $\max_{\ell\in\mathcal{L}} \pi_\varphi|_{\ell'} > \pi_\varphi|_\ell + \eta_\varphi$, i.e., the gain from relocating exceeds the moving costs. We use very small positive moving costs in our analysis. The reason for using small positive moving costs (on top of being obviously a realistic feature) is that: (i) we do not want firms to be glued to locations, i.e., multiplicity of equilibria is not driven by the existence of large moving costs; but (ii) we do not want firms to cycle between locations when there are tiny profit differences. Recall that all agents are indivisible in this model, so there are cases where tiny profit differences cannot be arbitraged away (by having fractional firms moving around as would be the case in a continuous model), so that we can have limit cycles. This problem obviously gets worse when there are large 'atoms' in the distribution of firms as is the case in our model. Since our positive assortative matching can produce quite skewed firm-size distributions (again a realistic feature), we need to cope with that in the numerical implementation. The bottom line is that we use small positive moving costs to find equilibria but that our results are not driven by the existence of these small frictions.

1.5 Equilibrium

An equilibrium is a distribution of firms and an assignment of workers to firms, such that: (i) no firm can profitably change location conditional on its employees, the locations of the other firms, and rents; (ii) no firm can profitably lay off workers; and (iii) no worker can profitably switch to another firm, given the distribution of workers across firms. Finally, real estate markets in all locations clear, i.e., aggregate demand for real estate in each location (from residents and firms) equals aggregate supply:

$$\sum_{\omega\in\ell} \frac{\alpha_\omega w_\omega^{\text{net}}(\mathbf{r})}{r_\ell} + \sum_{\varphi\in\ell} \xi|N_\varphi(\mathbf{r})| = \overline{L}_\ell \widetilde{D}_\ell r_\ell^{\frac{1-\beta}{\beta}}, \tag{14}$$

where $\mathbf{r} = (r_1, r_2, \ldots, r_{\mathcal{L}})$ is the vector of rents in all locations $\ell = 1, 2, \ldots, \mathcal{L}$. Observe from (14) that the equilibrium rents are interdependent via the assignment of workers to firms.

In extensions that allow for residential mobility of workers, we would also have the additional condition that: (iv) no worker can profitably change location conditional on its employer, the location of the other residents, and rents. In the case of worker movements, we again assume that the employer-employee relationship is maintained during the movement. The firm

is indifferent as to the worker's location, and the worker may after changing location search for a new employer if that is beneficial.

1.6 Endogenous transportation network

Cities may develop transportation networks. A network requires to pay a fixed setup cost; a cost per station (node) of the network; and a per kilometre cost of building links between nodes. The network is financed by a wage and profits tax that is levied on all agents in the initial equilibrium. The network is then constructed using the following procedure. First, we compute all bilateral commuting flows between locations in the city. The link with the largest flow seeds the network, i.e., locations ℓ and j with the largest flow get the first link and the first two nodes. Then, we look at the second largest flow for developing the second link. We do not allow for a new station to be built at less than some threshold distance (here, 1 kilometre) from an already existing station. We also check whether the new link is cheaper to build by attaching it to the closest existing node of the network rather than by building a new direct link. This procedure is repeated as long as budget remains. If the cost of a link exceeds the remaining budget, we move to the next largest positive commuting flow until either no more budget is available or all links with positive flows have been checked (and potentially build).

Once a network is developed (recall this need not be the case), agents optimize by choosing whether to commute directly or use the network. In other words, there is modal choice. Without a network, agents move along the straight-line distance to firms. Let $d_{\ell j}$ denote the distance between locations ℓ and j. In the presence of a network, the distance is

$$d^N_{\ell j} = \min_{n_1 \in \mathcal{N}} d_{\ell, n_1} + \min_{n_2 \in \mathcal{N}} d_{n_2, j} + \zeta \times \text{shortest path}(n_1, n_2), \tag{15}$$

where shortest path(n_1, n_2) is computed using a recursive implementation of Dijkstra's shortest path algorithm. The parameter $0 < \zeta < 1$ captures the travel time gains once on the network (if $\zeta \geq 1$, moving on the network is by construction never profitable). As seen from (15), access to and from the network is straight-line, and then there is a gain from moving along the network (distance on the network is equivalent to only ζ times distance off the network). Of course, agents choose the network if and only if $d^N_{\ell j} < d_{\ell j}$, and they travel off the network otherwise.

2 Technical aspects

The foregoing section describes the underlying economics of the model. We now turn to more technical considerations required for implementing the model. The source code has been written in `C++11` using an object-oriented approach. It has been developed using Apple's `X-Code 9.2` and partly draws on the `GNU Scientific Library 1.14`. The underlying code—which we will make available at a later stage of the project—will be part of the 'agent-based

urban simulation engine' (ABUSE) package that we presently develop. The objective of this package is to provide an environment that will allow to develop flexible urban simulation models along the line of the one we use in Ahlfeldt et al. (2020). The ultimate goal is to develop efficient code that will allow to manage relatively large problems with multiple dimensions of heterogeneity. The code we have used for now in Ahlfeldt et al. (2020) is a first non-optimized version that allows to simulate 'small' models.[2]

We now discuss a number of technical issues we need to make decisions on for implementing the ABM.

2.1 Sequentiality of moves

As in all agent-based models, there is path dependence and the sequence of moves of the agents matters. As explained before, we pick the initial assignment of workers to firms in an 'efficient' way by proceeding sequentially in decreasing order of worker and firm productivity. This yields the initial assignment of workers to firms. We then clear the real estate markets and allow for workers to rematch with firms (the matching can change when rents change to clear the housing/office market). This is again done in decreasing order of productivity, updating the rankings of workers in firms while doing so.[3] To keep the model computationally tractable, we clear the housing market only after each complete iteration, i.e., once each worker has been given the opportunity to make (or to not make) a move between firms. There are no severance frictions in the model, though these could easily be incorporated. Note also that some workers may remain unemployed if they cannot find any beneficial match.

Because of indivisibilities, there may be limit cycles where a very small share of workers want to change location once rents re-clear the markets (they move, conditional on rents; then new rents are determined, and they again want to move). To deal with this, we iterate the model until less than $\varepsilon_M \geq 0$ percent of workers move between firms. In a nutshell, we look at an equilibrium assigment up to a small fraction ε_M. The solver starts with $\varepsilon_M = 0$ and in case no stable assignment is found, that value is gradually increases. We keep track of the precision in the model output and can control for it. Most of the time, we can find an stable assignment for all workers. When that is not the case, we can control for the precision in the assignment in Tables 3 and 4 of Ahlfeldt et al. (2020), and the results do not change. We can also just drop all simulation runs where the equilibrium is not 'precise enough'.

Firm movements are implemented in a similar way. Starting with the most productive firm, each firm searches over locations to find the best location conditional on its current worker

[2]For example, we need more systematic and efficient code to deal with the assignment of workers to firms. With multiple uncorrelated dimensions of heterogeneity, the assignment is not strictly positive assortative, which complicates the search for an assignment.

[3]We a worker leaves a firm, her rank becomes vacant and all other workers with higher rank move down in the hierarchy. The worker who leaves joins the new firm at the highest rank.

assignment, rents, and agglomeration externalities. Once all moves are done (the updating is simultaneous), we allow for rematching of workers and housing market clearing as described before.[4] We iterate until less than ε_S percent of firms change location. Again, most of the time we find an exact equilibrium and we can control for precision.[5]

2.2 Shocks

We model shocks in a simple way as affecting firms in the historic prime location (i.e., the location with the largest share of employment in the initial equilibrium). There is a random shock that hits between 50–100% of firms in the prime location and forces them to move to a different location (think about this as part of the prime location being destroyed). We then let the model settle into a new post-shock equilibrium, where only the displaced firms pick new optimal locations and where all labor relationships and rents are updated. However, we hold at this stage the locations of the remaining firms constant (we think about the shock as affecting only a subset of firms temporarily).

Once this is short-run adjustment is completed, the shock is resorbed, i.e., all firms can re-optimize (and the worker assignment and rents are updated) and the initially shocked location becomes again available as a potential location of choice for all firms. We then find the new long-run equilibrium where all firms can re-optimize, and all workers can be reassigned to firms while housing markets clear.

3 Some implementation details

Below is an outline of the classes that we use. The key ingredients are locations and agents. Agents can be of three types: (i) workers; (ii) firms; and (iii) administrations (e.g., the city hall in the model). They are all derived from a base class that specifies common behavior (and which allows for polymorphism). Geographic coordinates are modeled using lat-lon points.

At this stage, all relationships between objects are modeled using (smart) pointers. Future versions of the ABUSE engine will replace those by identifiers to make the code more transparent and user-friendly.

[4] Clearing the housing market after each firm moves is not feasible. For example, in our numerical implementation, there are 102 locations so that we need to solve a system of 102 non-linear equations. With 1000 firms, we would need to solve 1000 such systems for each iteration of firm moves, which is not feasible.

[5] As explained before, in an ideal run, ε_M and ε_S would both be always zero. Yet, given indivisibilities, the many-to-one assignment, and the non-linear equation system for clearing the housing market, we can small limit cycles where a few workers jump between firms or firms jump between locations. These can be mostly ruled out using small moving frictions. We keep track of the 'precision' of the assignment and of the location equilibrium in all model runs. Excluding runs that show slightly less precision does not affect in any way the results we present in this paper.

```
class point;
class location;
class agent;
class worker : public agent;
class firm : public agent;
class admin : public agent;
class directed_graph;
struct edge;
```

Housing market clearing uses the GSL `gsl_multiroot_fsolver` implementation with parametrization `gsl_multiroot_fsolver_hybrid`. We force the solver to stay away from boundaries, i.e., we reset it when rents fall below 0.00000001 or exceed 10000000 (we refer to this as 'backtracing'). We also keep track of the initial rents. Should we not be able to converge to a solution, we perturb the old initial point and solve the model again using the new starting point.

Output for the initial allocation, the initial equilibrium, the short-run equilibrium after the shock, and the final long-run equilibrium is written to files. This output comes in two flavors: (i) aggregate output on the simulation run such as the geographic concentration measures (this underlies Tables 3 and 4 in Ahlfeldt et al., 2020); and (ii) detailed output at the level of locations and agents (which allows us to compute location-specific and matching-specific characteristics; this underlies much of the output in Appendix A.5 in Ahlfeldt et al., 2020).

For computing the shortest path on the network, we have adapted Dijkstra's shortest path algorithm from Michal Forisek `https://www.quora.com/What-is-the-most-simple-efficient-C++-code-for-Dijkstras-shortest-path-algorithm`. This version of the algorithm uses and efficient recursive implementation that can be adapted to the way we structure our `edge` and `directed_graph` classes.