Alternating Stutter Bisimulation

Jonas Krook, Robi Malik, Sahar Mohajerani, Martin Fabian
krookj@chalmers.se, robi@waikato.ac.nz, mohajera@chalmers.se, fabian@chalmers.se

Defining a partition

For an equivalence relation $\mathcal{R} \subseteq S \times S$, the equivalence class of $s \in S$, denoted $[s]_\mathcal{R}$, is the set $\{s' \in S \mid (s, s') \in \mathcal{R}\}$. The equivalence classes of $\mathcal{R}$ form a partition of S, wherein they are referred to as blocks. We call the union of any number of equivalence classes for a superblock, and the set of all superblocks of $\mathcal{R}$ for $\mathrm{SB}(\mathcal{R})$.

A transition system is a tuple $G = \langle S, \Sigma, \delta, S^\circ, \mathrm{AP}, L\rangle$ where S is a set of states; Σ is a set of transition labels; $\delta \subseteq S \times \Sigma \times S$ is a transition relation; $S^\circ \subseteq S$ is a set of initial states; AP is a set of atomic propositions; $L\colon S \to 2^{\mathrm{AP}}$ is a state labelling function.

A path fragment of G is a sequence of states $\pi = s_1s_2s_3\ldots \in S^*$ such that $(s_i, \sigma, s_{i+1}) \in \delta$ for some $\sigma \in \Sigma$ for all i.

We say that $\langle G, s_1\rangle \vDash [s_1]_\mathcal{R}\, \mathcal{U}\, T$, for $T \in \mathrm{SB}(\mathcal{R})$, if there exists an $i > 0$ for each infinite path fragment $\pi = s_1 \ldots s_i s_{i+1}$ such that $s_j \in [s_1]_\mathcal{R}$ for all $j \leq i$ and $s_{i+1} \in T$. Furthermore, $\langle G, s_1\rangle \vDash [s_1]_\mathcal{R}\, \mathcal{W}\, T$ allows also paths with $s_i \in [s_1]_\mathcal{R}$ for all $i > 0$.

A controller for G is a function $C\colon S^+ \to 2^\Sigma$. A positional controller is a function $\bar{C}\colon S \to 2^\Sigma$. The transition system resulting from controlling G by C is denoted C/G.

Let $\mathcal{R}$ be an equivalence relation over S and let $(s, t) \in \mathcal{R}$. $\mathcal{R}$ is an alternating stutter bisimulation iff

(i) $L(s) = L(t)$

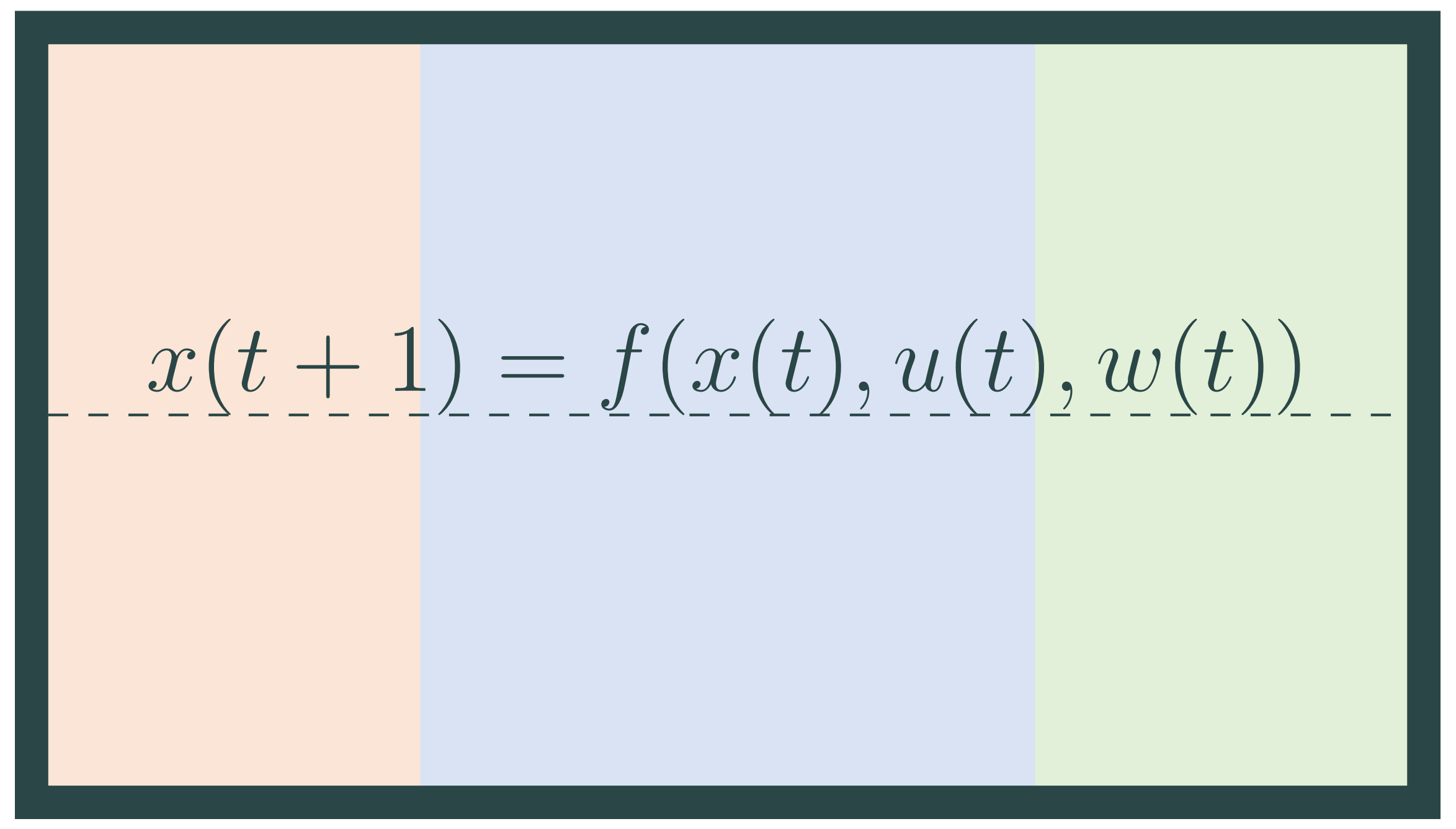
(ii) if, for some positional controller, $\langle \bar{C}_s/G, s\rangle \vDash [s]_\mathcal{R}\, \mathcal{U}\, T$ for some $T \in \mathrm{SB}(\mathcal{R})$, then there exists a positional controller $\langle \bar{C}_t/G, t\rangle \vDash [s]_\mathcal{R}\, \mathcal{U}\, T$

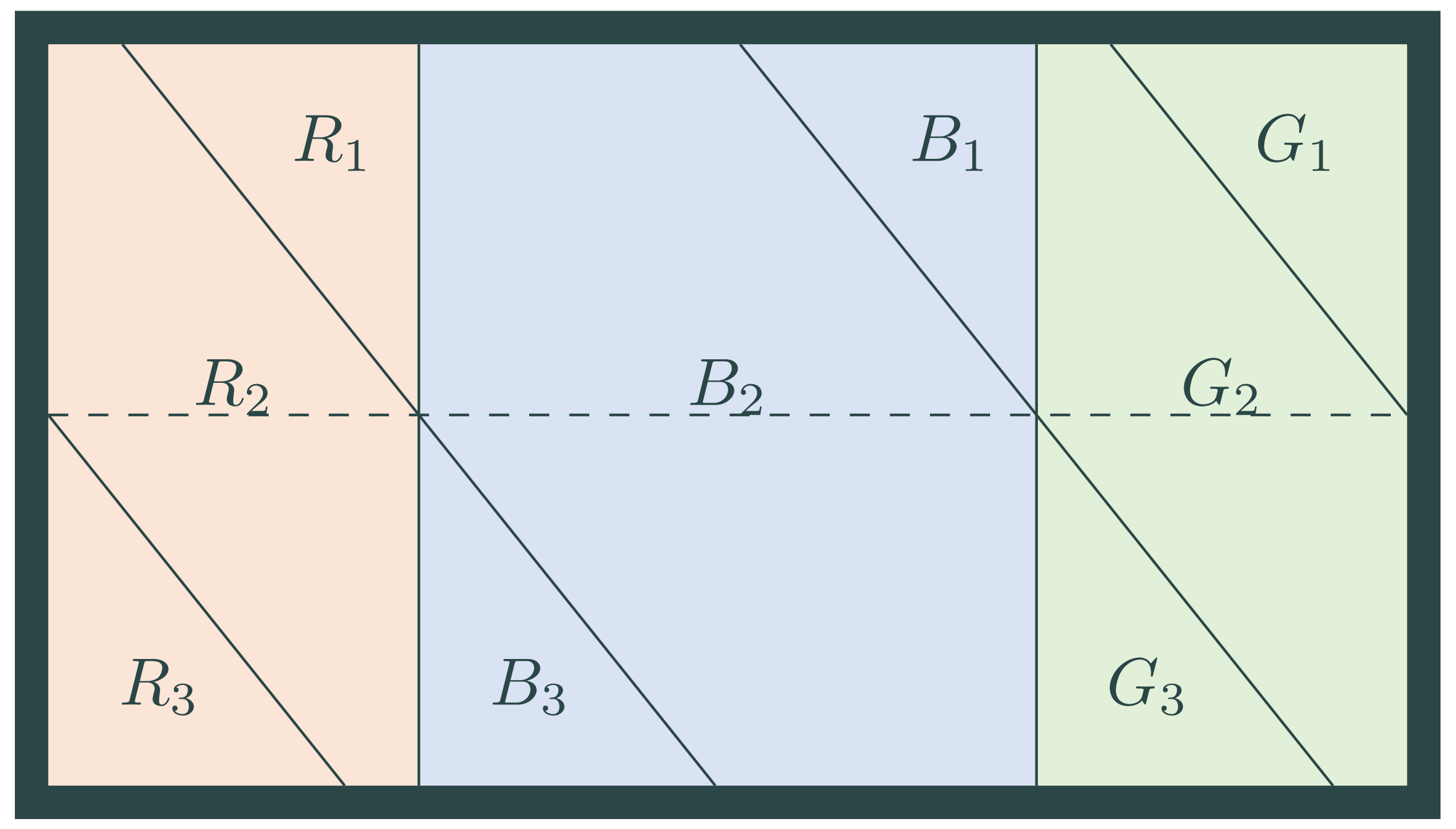
(iii) same as (ii) but with $\mathcal{W}$.

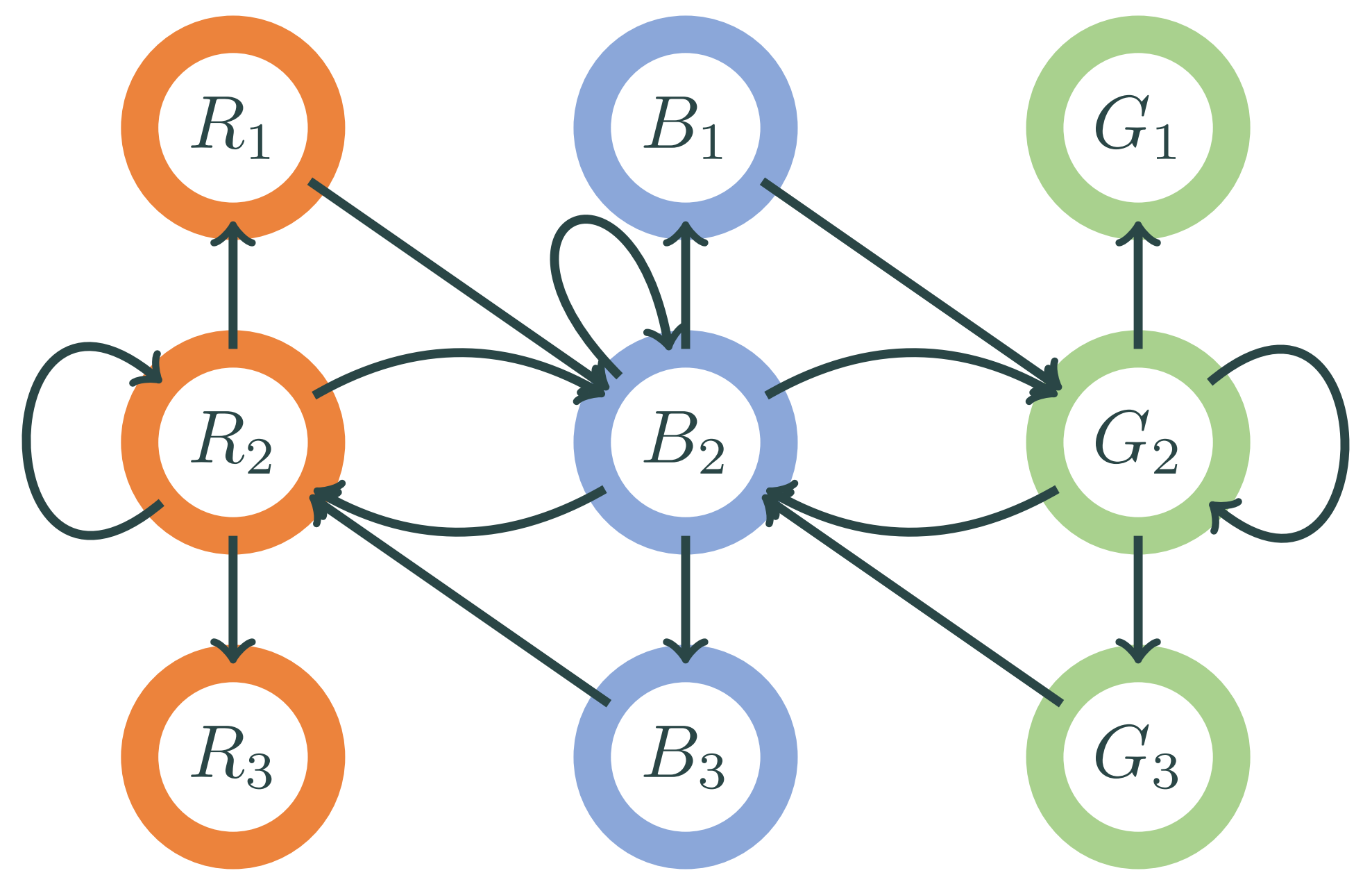
Abstraction that preserves $\mathrm{LTL}_{\setminus\circ}$ specifications under robust control

We want to use a fragment of Linear Temporal Logic (LTL) without the next operator to specify safety-critical requirements and synthesize a robust controller that fulfills those specifications on a discrete-time continuous state system that is subject to disturbances. Essentially, the robust controller must ensure that specific subsets of the state space are visited in an order which is allowed by the formal $\mathrm{LTL}_{\setminus\circ}$ specification. These subsets are represented by different color labels. For instance, we might want to implement a controller that forces the system to visit the red and green subsets infinitely often.

However, the synthesis method cannot be applied directly on a continuous state space because it takes finite-state transition systems as input. One way to bridge this gap is to divide the continuous state space into a finite partition and let each block of the partition be one state in a transition system, which is called an abstract system. We use the equivalence classes of an alternating stutter bisimulation as the partition. The transitions in the abstract transition system are then based on how the original, or concrete, dynamical system can be robustly controlled within and between the blocks.



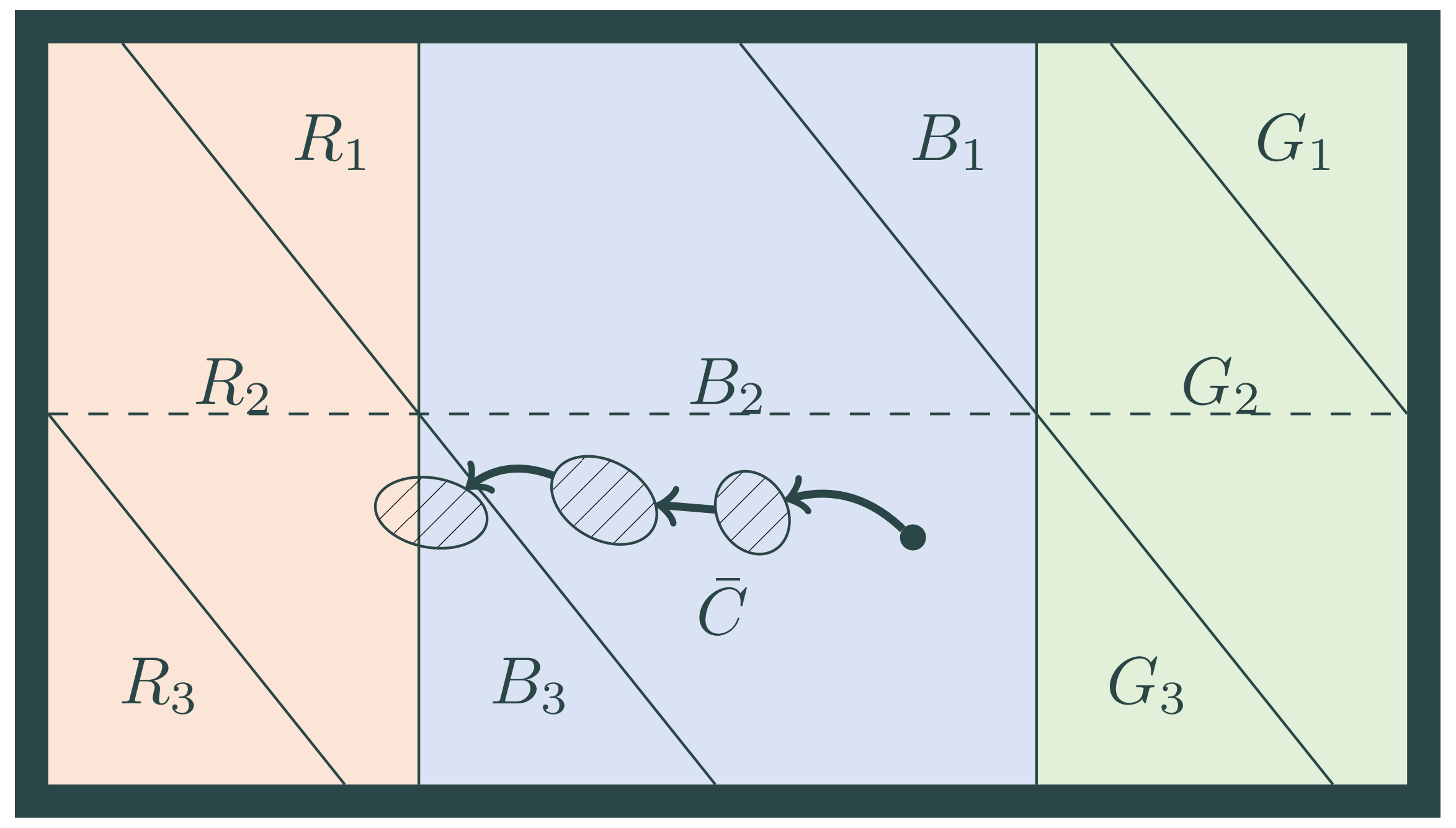




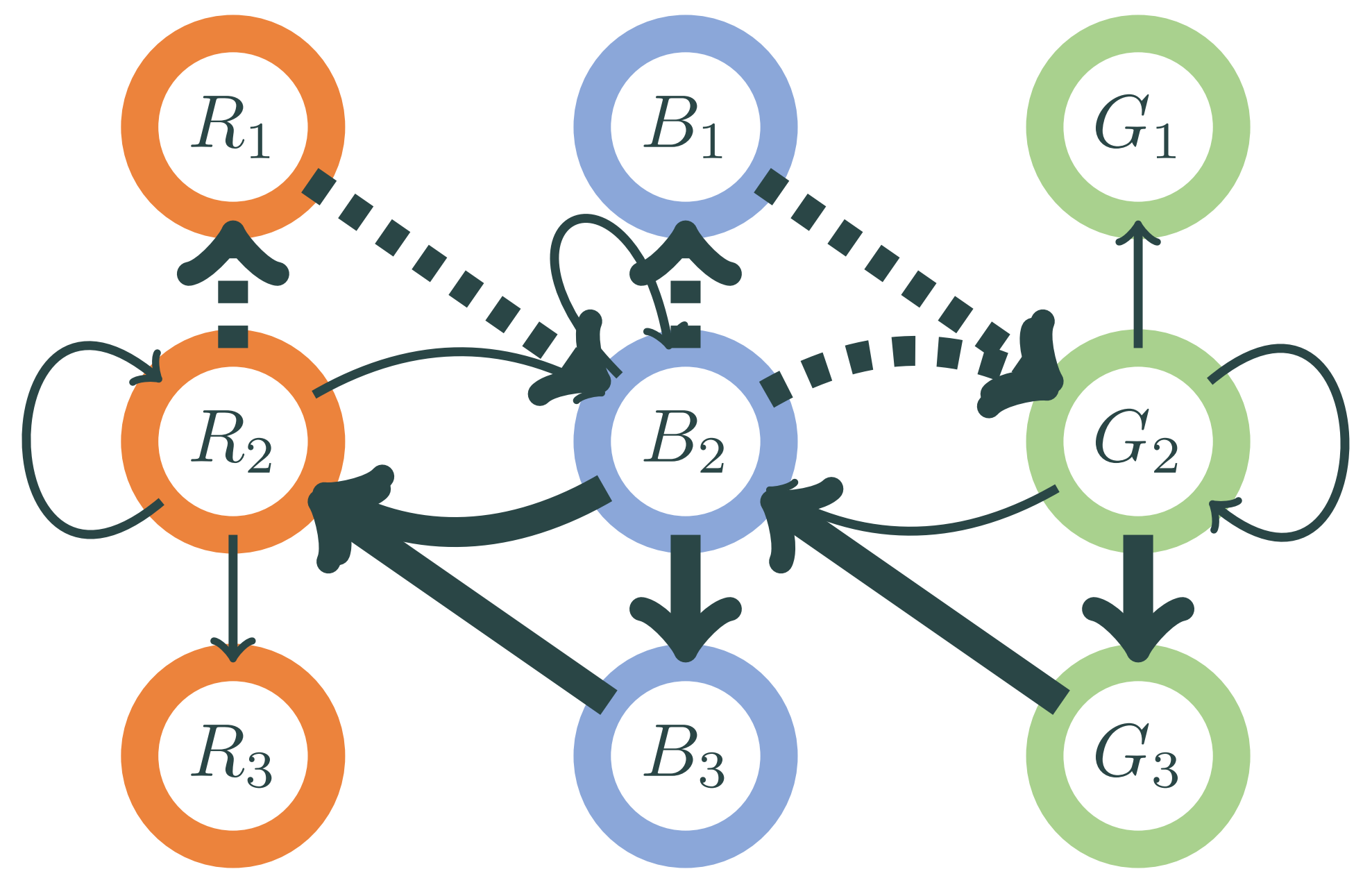
An abstract controller for the abstract system decides on control actions in the form of sets of allowed transitions, and the disturbance or process noise determines which of these allowed transitions are taken. The choices available to the abstract controller in an abstract state are based on the existence of concrete robust positional controllers that can robustly control the concrete system from the states in the corresponding source block to states in a target superblock. The blocks (equivalence classes) of alternating stutter bisimulations are defined in such a way that the system can be controlled to the same target superblocks from any of the source block's states.

For instance, if a concrete positional controller (e.g. $\bar{C}$) can control the system from one state in block B_2 to a set of states in the superblock $B_3 \cup R_2$, then an abstract controller can choose the abstract states B_3 and R_2 as the next possible states from abstract state B_2. This works since it must exist concrete controllers for any state in B_2 that can control to $B_3 \cup R_2$.

A self-loop is added to an abstract state if there exists a concrete positional controller that lets the concrete system remain in the corresponding block forever.



We can now synthesize a controller for the abstract system such that the red and green abstract states are visited infinitely often. Every abstract control action has a corresponding concrete controller forcing the transition, so a concrete controller fulfilling the requirement can be implemented as a sequence of concrete positional controllers.


R_1
B_1
G_1
R_2
B_2
G_2
R_3
B_3
G_3