Session III: Interacting with Hardware

Bridge the gap

- Between Bloqade and hardware!

$$\frac{H}{\hbar} = \sum_i \frac{\Omega(t)}{2}\left(e^{i\phi(t)}|g_i\rangle\langle r_i| + e^{-i\phi(t)}|r_i\rangle\langle g_i|\right) - \sum_i \Delta_\mathrm{i}(t) n_i + \sum_{i<j} V_{ij} n_i n_j$$

`BloqadeSchema.submit_to_braket`

`emulate!(prob)`

Learning objectives

By the end of this class, you will be able to:

- **Describe** the Bloqade to Hardware pipeline
- **Differentiate** transformation and validation functions to work within Hardware Constraints
- **Design, Submit, and Retrieve** Hamiltonians for Hardware

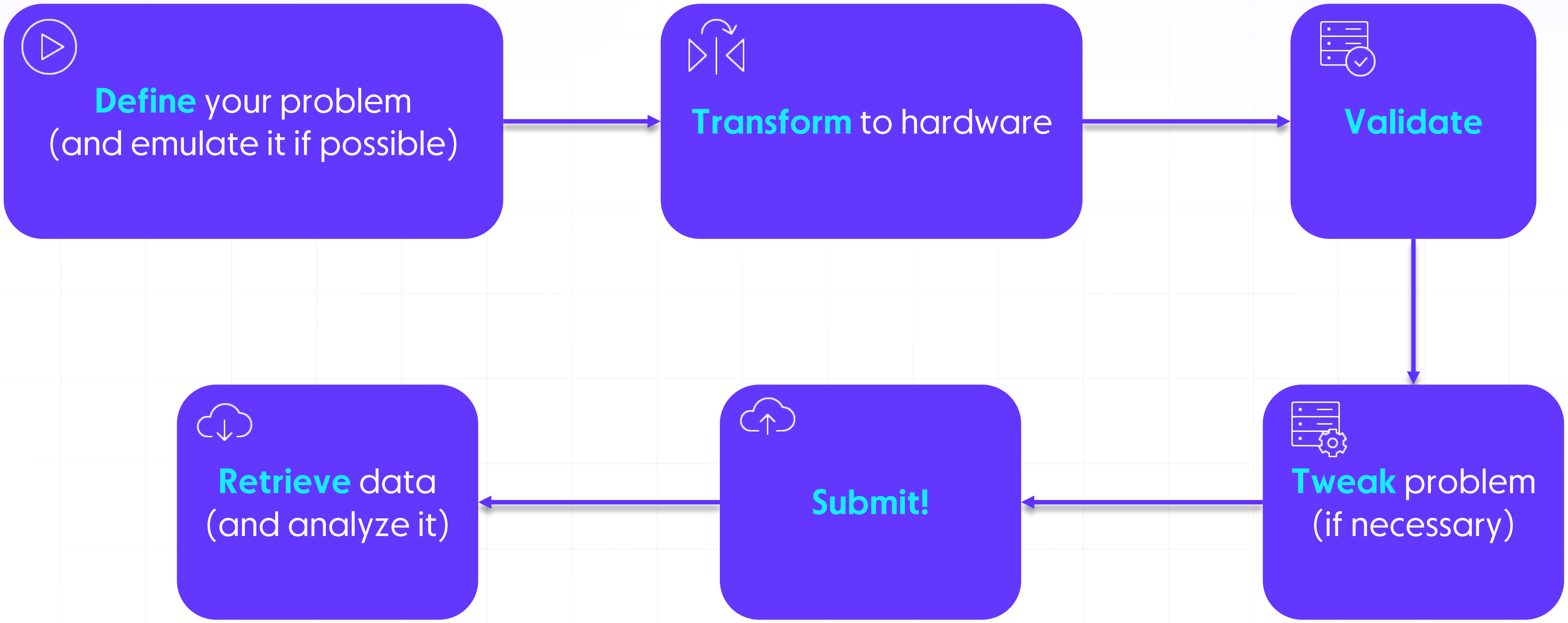
Start with a question:

- If you got here after going through sessions I and II, you already know how to operate Bloqade for doing emulations.

  - Activity: Think-share

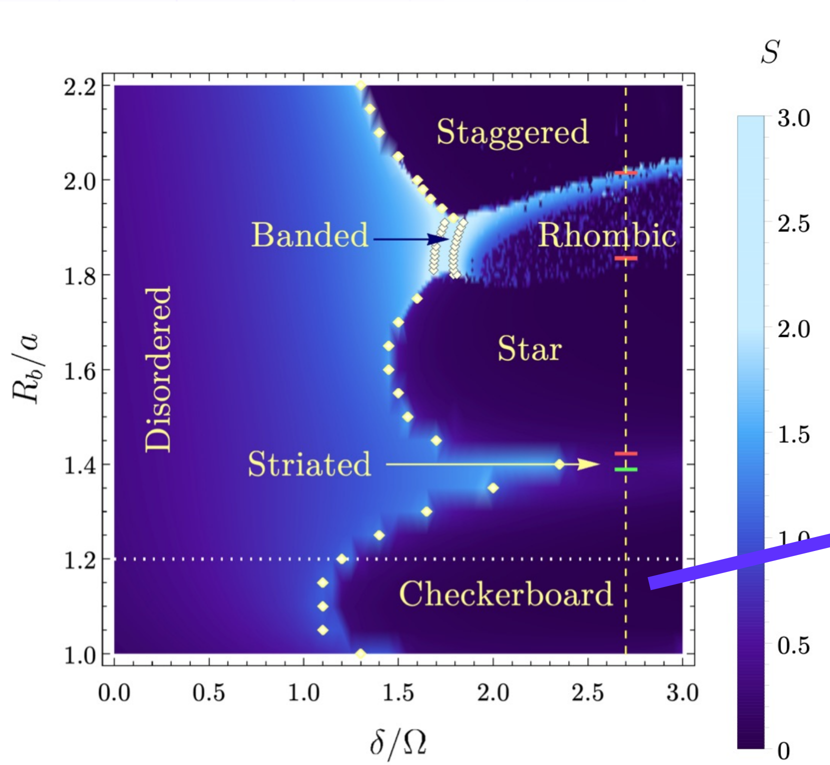
    - What steps would you claim need to happen in order to submit an algorithm you developed in Bloqade to run in a real quantum computer?

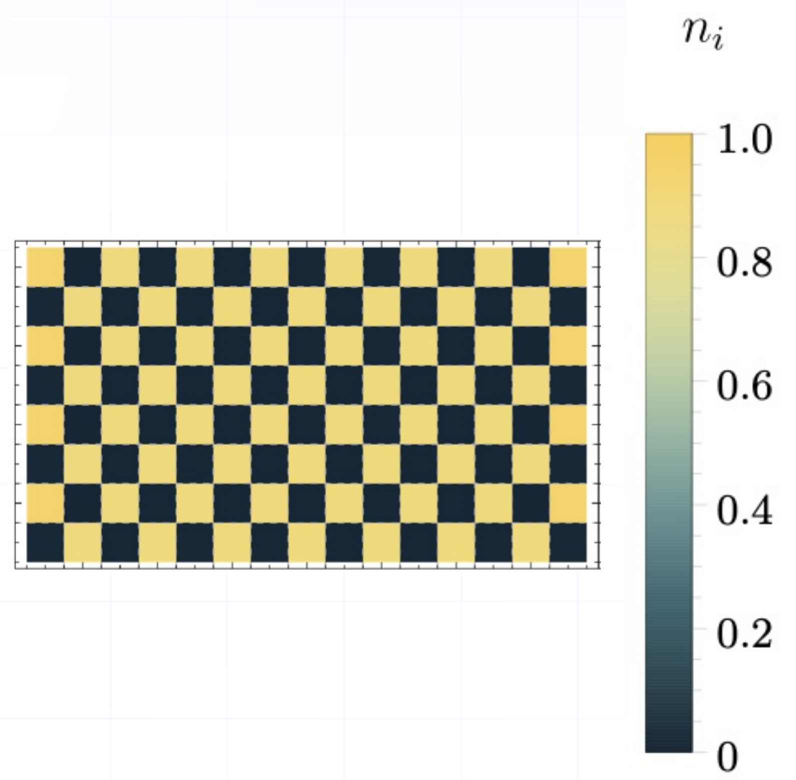
Big picture


Define your problem
(and emulate it if possible)
Transform to hardware
Validate
Tweak problem
(if necessary)
Submit!
Retrieve data
(and analyze it)

Start With a Problem

- *Attempt to recreate **2D Checkerboard Phase***





Let's revise on Bloqade!

R. Samajdar et al., Phys. Rev. Lett. 124, 103601 (2020)

|QuEra>

`transform` and `validate`: why?

Make sure your algorithm will run BEFORE submission to hardware

transform

- *Hardware has some important limitations to consider*

Activity: name some hardware constraints you learned of in previous sessions

- Atoms have a minimum distance they can be placed next to each other
- Have finite-valued Rabi, Detunings, and Phase
  - Final waveforms on hardware must be piecewise Linear/Constant (discretization necessary)
  - Your slope or “slew rate” cannot exceed certain maximums
- Minimum time resolution to consider (smallest increment of time you can define)

Find Hardware Constraints documented Here:
https://queracomputing.github.io/Bloqade.jl/dev/capabilities/

Global Rydberg Values

Capability	Field	Value
Rydberg Interaction Constant	`c6_coefficient`	5.42×10^6 rad/µs × µm^6
Minimum Rabi Frequency	`rabi_frequency_min`	0.00 rad/µs
Maximum Rabi Frequency	`rabi_frequency_max`	15.8 rad/µs
Rabi Frequency Resolution	`rabi_frequency_resolution`	0.0004 rad/µs
Maximum Rabi Frequency Slew Rate	`rabi_frequency_slew_rate_max`	250.0 rad/µs^2
Minimum Detuning	`detuning_min`	-125.0 rad/µs
Maximum Detuning	`detuning_max`	125.0 rad/µs
Detuning Resolution	`detuning_resolution`	2.0×10^{-7} rad/µs
Maximum Detuning Slew Rate	`detuning_slew_rate_max`	2500.0 rad/µs^2
Minimum Phase	`phase_min`	-99.0 rad

Objective of `transform`

MAXIMIZE *your flexibility to design algorithms*

MINIMIZE *constraint bookkeeping by hand*

validate

- Occasionally, there are certain things that can’t be automatically transformed
  - Most commonly with atom position constraints
- Require some form of user intervention, this is where validation is necessary!
- Treatable as a catch-all for any incompatible Hamiltonians

Submitting

- *If the Hamiltonian passes Validation, you'll need your AWS Credentials to submit*
- *Upon submission, may need to wait a bit as tasks go on a queue that the machine will consume from when it's open*

Tuesdays	16:00:00	to	20:00:00	UTC
Wednesdays	16:00:00	to	20:00:00	UTC
Thursdays	16:00:00	to	18:00:00	UTC

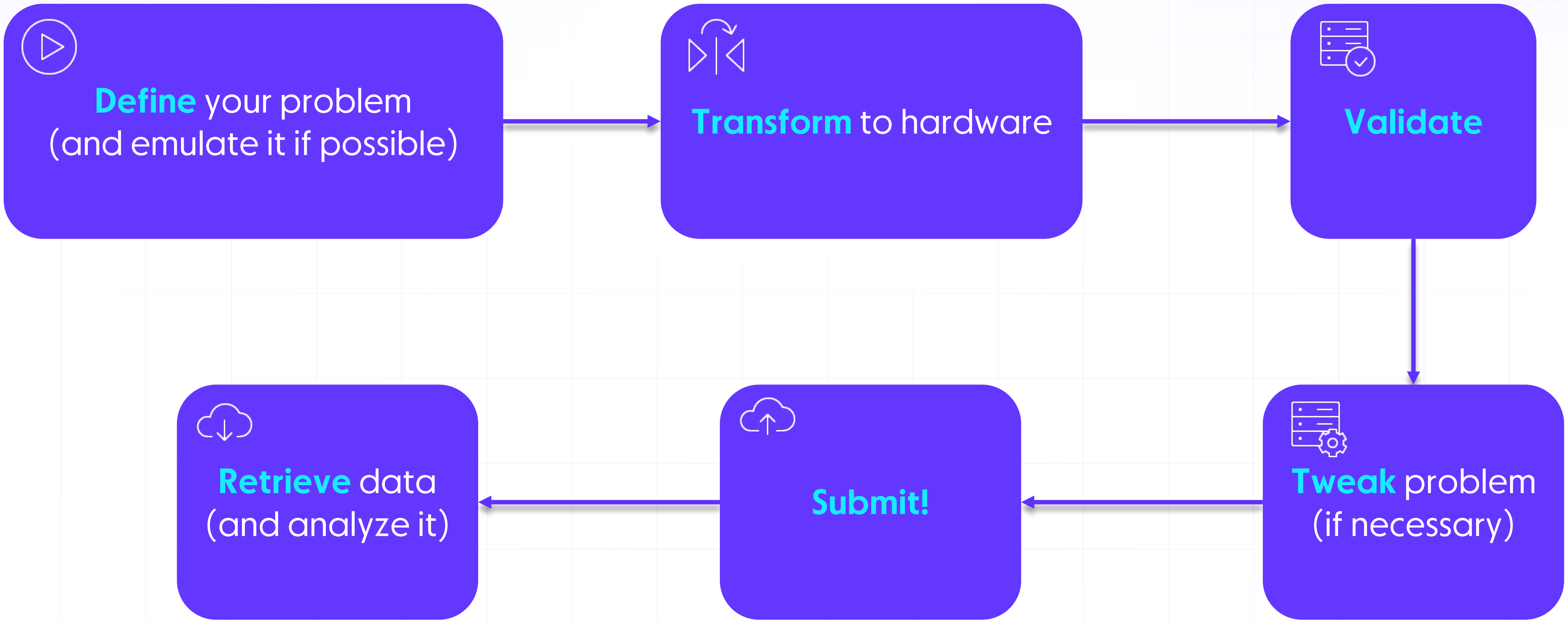
*Outside of hours, tasks can be submitted to the Amazon Braket queue

|QuEra>

Retrieval

- Heavy lifting done by Braket.jl
- Results can be saved in HDF5-Compatible format or JSON for usage inside Python
    - JSON is the friendlier format!

Summary


Define your problem
(and emulate it if possible)
Transform to hardware
Validate
Tweak problem
(if necessary)
Submit!
Retrieve data
(and analyze it)

Learning objectives

Now you are able to:

- Describe the Bloqade to Hardware pipeline
- Differentiate transformation and validation functions to work within Hardware Constraints
- Design, Submit, and Retrieve Hamiltonians for Hardware