Multi source cooling control algorithm

Luca Bortot
ENI
Italy
luca@bortot.it

Walter Nardelli
ENI
Italy
walter.nardelli@eni.com

Peter Seto
EE HPC WG
USA
jordanruthseto@aol.com

***Abstract*— This paper describes how the automation team at ENI Green Data Center (GDC)[1] solved the problem of controlling the temperature inside a data room that can be air cooled both with direct free-cooling and with water-based chillers and heat exchangers using a sliding mode control design. They created an algorithm which pro-vided: 1. a directional, incremental "walking " function to seek out and test alternative solutions, biased toward a preferred solution, 2. stabilization on an optimized solution, 3. a destabilization function to retest the solution periodically without allowing jitter, and 4. a meth-od of changing preference to autonomously reoptimize for another configuration.**

I. INTRODUCTION

Automating the cooling system(s) in data centers has been a challenge. Ambient air enthalpy varies (weather) and computer heat generation (load) can vary signifi-cantly, so cooling by varying the mix of ambient air, recirculated computer room air, tower-cooled water, and compressor-chilled water requires human oversight and constant experimentation to execute. However, many data centers operate with little or no staff for part of each day and over weekends. Current practice in HPC uses industrial Air Handling Units (AHUs) to condition ambient air using software offered by build-ing management system (BMS) vendors. [1] Outcomes are often unsatisfactory because most BMSs are de-signed for far lower heat loads with less fluctuation than those typical in HPC facilities. AHUs require com-plex programming to map input and output control variables to adapt to large ranges of conditions. They also can trap cooling capacity. [2]

HPC centers are incorporating PID and PLCs which are standard in automated industrial designs. Augmented instrumentation, also called Operational Data Analytics (ODA), is being added in HPC centers which allows fine tuning of setpoints for cooling energy minimization [3]. State of the art practices of designing cooling systems for HPC using setpoints based on vendor supplied val-ues has been shown to create trapped and stranded ca-pacity which grow with the growing energy demands of peta-to-exa scale systems. Use of ODA to measure the temperature of components in the rack has made using preset input setpoints less desirable [3] . The need to create automated controls which are agnostic (not based on a predetermined and fixed mapping of the cooling system) which can incrementally adjust set-points and employ components based on fed-back sys-tem responses prompted the designers at ENI's GDC to create a version of "sliding mode control" in their con-trol software. The limitations of sliding mode required adaptation to allow its use in the GDC [7]. To do this they first needed to reduce the complexity of the cool-ing system, composed of many very different compo-nents operating both in tandem and in parallel with each other to a single scalar control input value, and a single control output value as required by the logic of PID control.

II. GDC COOLING FUNDAMENTALS

The GDC is air-cooled, primarily free-cooled, but has a compressor-based cooling plant to supply chilled water if required. Its design incorporates the features of an AHU into the architecture of the building itself.

Given the amount of energy and the thermal inertia of the cooling air mass, control algorithms designed for industrial AHU use proved unable, in practice, to main-tain the setpoint in every operating condition with a maximum allowed error of 3C without human inter-vention, as required at GDC, and the control logic had to be developed from the ground up.

A. Operating modes of the GDC.

Figure 1 depicts the data room devices involved in temperature control[2], together with arrows showing the expected airflow.

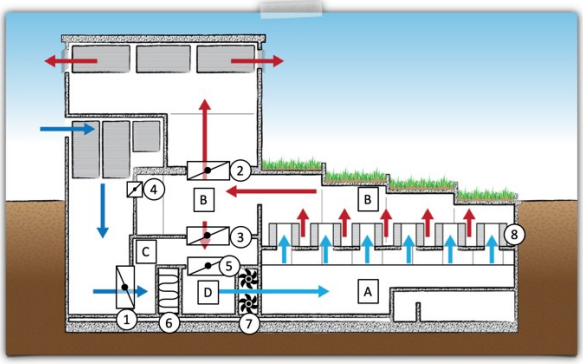


Figure 1: Devices and airflow

1– Inlet damper 2- Exhaust damper 3- Recirculation damper 4- Anti-icing damper 5- Bypass damper 6- Heat exchangers[3] 7- Turbines 8- Computers

A- Plenum B- Hot aisle C- Mixing room D- Fan room

[1]The GDC hosts HPC4, ENI latest HPC system, currently ranked #17 on the Top500 List

[2] This schema includes all GDC dampers, although 4 and 5 are not relevant for temperature control and will be therefore ignored for the remainder of the paper (the reader can consider them closed).

[3] Heat exchangers are Cu/Al cooling water-coils, with an internal volume of 309l and an exchange surface of 1228m^2. There are 8 of those per data room, with a possible upgrade to 16.

B. *Operating modes*

In order to maximize energy efficiency, the data room can be configured in three different operating modes.

1) *Total free-cooling (economizer)*

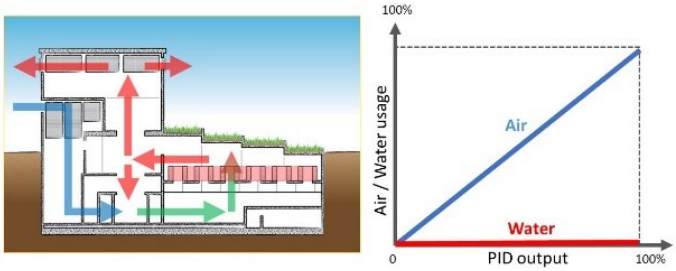


Figure 2: Total free-cooling airflow

In total free-cooling only external air is used. This is the expected mode when outside weather is cool enough. Dampers and water valve setup is expected to be as follows:

- Inlet damper (1): modulating
- Exhaust damper (2): modulating
- Recirculation damper (3): modulating
- Water valve (6): closed

2) *Partial free-cooling control*

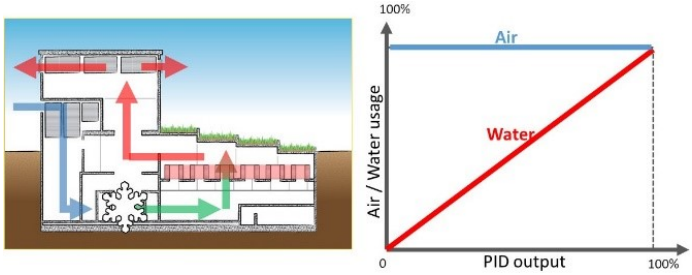


Figure 3: Partial free-cooling airflow

In partial free-cooling, air is used together with water[4]. This mode is used when outside air is not cool enough for total free-cooling but still cooler than the Hot Aisle (B). Dampers and water valve setup is expected to be as follows:

- Inlet damper (1): full open
- Exhaust damper (2): full open
- Recirculation damper (3): closed
- Water valve (6): modulating

3) *Recirculation*

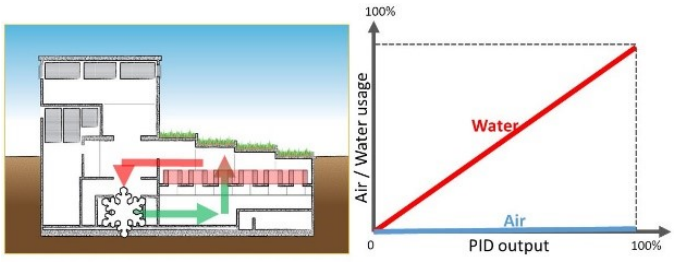


Figure 4: Recirculation airflow

In recirculation there is no use of outside air, only water. Dampers and water valve setup is expected to be as follows:

- Inlet damper (1): closed
- Exhaust damper (2): closed
- Recirculation damper (3): full open
- Water valve (6): closed

There are various reasons for using this mode:

- *Energy saving*: when outside enthalpy is higher than data room's, recirculation gets cheaper than free-cooling (because the load on the chillers is lower)
- *Humidity control*: at the GDC we decided that expensive humidifiers/dehumidifiers were not cost effective: in the rare cases when humidity would go outside the allowed range, the automation software switches to recirculation and keeps humidity and temperature constant in the data room
- *Emergency isolation*: in case of environmental hazards

III. PID-CONTROLLED COOLING SYSTEM

The GDC control goals were to maintain setpoint (with 3C maximum allowed error), and use the most energy efficient cooling configuration possible.[6]

We decided to base the control system on a PID controller (proportional-integral-derivative), which is well established and understood and readily available in PLCs.

As a consequence *inputs are limited to a single (process) variable and an output (control) variable.*

The process variable is temperature, as the humidity is not actively controlled. (When it is about to exit the permitted range, the system switches to recirculation mode, keeping both temperature and humidity constant).

We consider as control variable (C, with values 0-100) the cooling demand of the data center (that can be met with air, water, or a mix of the two). C then needs to be translated into corresponding control variables for the air dampers and the cold water valve.

IV. REDUCING VARIABLES

The first problem to solve is how to reduce 4 control variables (3 dampers + water valve) to one, as required by the PID controller.[7]

As the total airflow in the data room can be considered constant the amount of air entering the system equals the amount leaving the same system. We can state that:

$$Inlet = Exhaust$$

where inlet and exhaust, in this context, are the dampers control variables whose values range from 0% (closed) to 100% (full open).

Similarly, the amount of air that is not leaving the system through the exhaust damper will recirculate:

$$Recirculation = 100\% - Exhaust$$

So the three dampers are linked together: the state of any one will determine the state of the others. If we use as reference the

[4] Cooled water temperature is variable depending on the current temperature setpoint. Typically it is 5C lower. As room temperature can vary between 18C and 28C, water temperature ranges between 13C and 25C (see more on[7], [10])

inlet damper, which represents "how much external air I'm using", we have an "Air" control variable so that:

$$Air = [0..100]\begin{cases} Inlet = Air \\ Exhaust = Air \\ Recirculation = 100 - Air \end{cases}$$

We have thus reduced three air dampers to a single Air control variable. This leaves us with 2 control variables: Air (as described above) and Water (water valve opening), both ranging between [0..100]. This is still not enough for a standard PID[5].

We then decided to handle both cooling sources at once, using a sliding mode control.

V. Sliding Mode Control

From the control point of view, any configuration of Air/Water that obtains the setpoint is satisfactory, but boundary conditions may make either Air or Water *preferred* over the other: whenever there are conditions for using Recirculation, Water is preferred, otherwise it's Air.

1. Primary and secondary cooling sources

Then there are two cooling sources, one of which is (at any given time) the *preferred*, now called the primary (P) source, and the other is the secondary (S). The goal is to meet the cooling demand as much as possible with the primary source and only add cooling from the secondary source when really needed.

The cooling demand as described by the control variable C obtained from the PID controller now needs to be translated into the corresponding values for the primary and secondary source. This process is best described by the visualization in Figure 5: 2D Control space: For example, a C-value of 50 could either translate to P-value of 100 and an S-value of 0, or by identical P- and S-values of 50. A C-value of 100, on the other hand, could only be reached by P- and S-values of 100. Any combination of P- and S-values is possible, as long as it follows the formula (P+S)/2 = C.

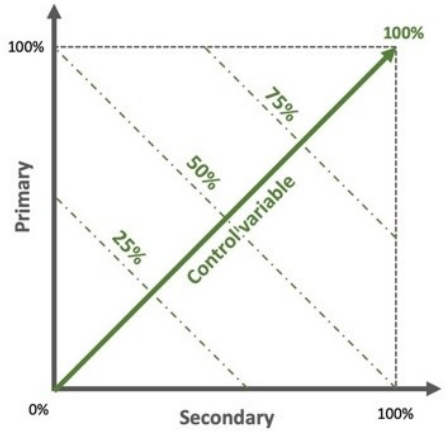


Figure 5: 2D Control space

For every value of the control variable there are infinite pairs of P and S, but we want the control to push the configuration towards the primary axis and away from the secondary axis.

A. The "thirds rule"

In each loop iteration, we must bias the response to changes in cooling demand (ΔC) to the primary: if more cooling is required, we add 2/3*ΔC primary and 1/3*ΔC secondary. If less is required, we reduce 1/3*ΔC primary and 2/3*ΔC secondary (this we call the "thirds rule").

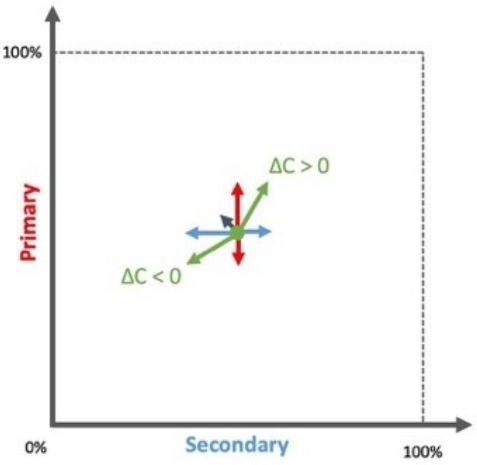


Figure 6: Thirds rule

The result of the thirds rule is a bias for the primary axis.

B. Axis stickiness

The configuration which maximizes P and minimizes S is the primary axis, but the control is always running, doing small adjustments at every loop; due to the thirds rule, even a minimal increase in cooling will detach the system from the axis towards S by (+1/3*ΔC).

The system will bounce off the optimal axis P.

This is a highly undesirable behavior, especially when P is Air and the weather is cold, because this means that we can never shut down the chillers – a minimal, intermittent water requirement is always present.

To avoid this we introduced the concept of "axis stickiness": if the control is on an axis, only a strong enough cooling demand (Thr_{sticky}) can "detach" the system from that axis; all lesser ΔC produces a movement along the axis itself.

[5] In every distinct operating mode only one variable is used, and this may suggest to build three distinct controls and switch from one to another when efficiency may benefit. This was tried, but resulted in a "mode change" that, given the size of the system, won't maintain the setpoint.

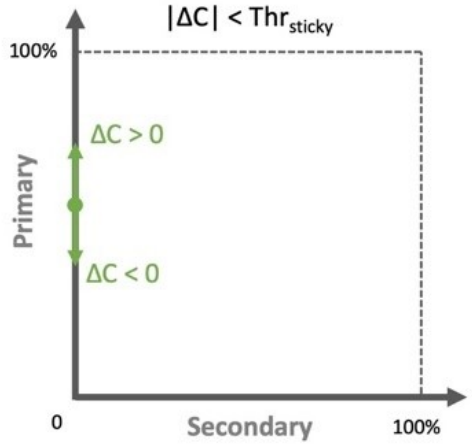


Figure 7: Stickiness applied

Figure 7 shows what happens when ΔC is below the stickiness threshold: the system stays on the axis despite the "thirds rule".

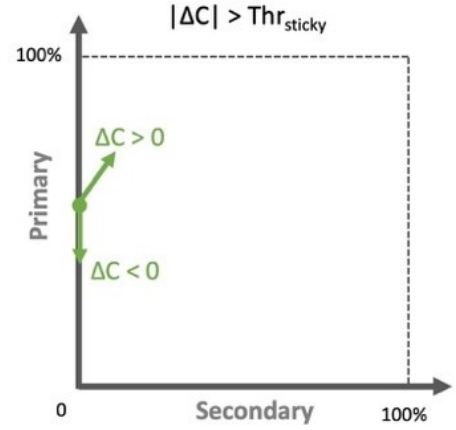


Figure 8: Detaching from axis

Figure 8 shows what happens if there is a strong cooling demand: stickiness no longer applies, and the control is free to use both P and S to quickly respond to a sudden increase in cooling demand.
In a stress test adding an 8MW instantaneous load, the system was able to use water immediately, initially going beyond 10% (the optimal configuration), in order to allow time for the air dampers to completely open. As the setpoint was attained, the "thirds rule" progressively reduced water down to 10% when air attained 100% (maximum over-temperature < 2.5C).

C. *Forced unbalancing*

Every real-world PID utilizes a "neutral zone" to prevent excessive wear of mechanical devices.
The neutral zone is the range of error on the process variable that is "tolerated". Within this zone, the PID does not change the control variable.
As the system approaches the setpoint and oscillations are damped, we do not want "jitter" micro-adjustments that only cause mechanical wear.
There is a drawback though: if the PID is in neutral zone and is not on an axis, the thirds rule can't be exploited (ΔC = 0!).

Also, because the "thirds rule" is suspended in the neutral zone, if we're on the primary axis in neutral zone and we reverse source preference, we instantly are moved to the secondary axis (worst configuration possible) and there's no PID activity to begin the move towards the (new) primary axis.
For these reasons we introduced the concept of "forced unbalancing":
While the PID is in neutral zone and S > 0 and P < 100, force P += U_q, S -= U_q (where U_q = unbalancing quantum)

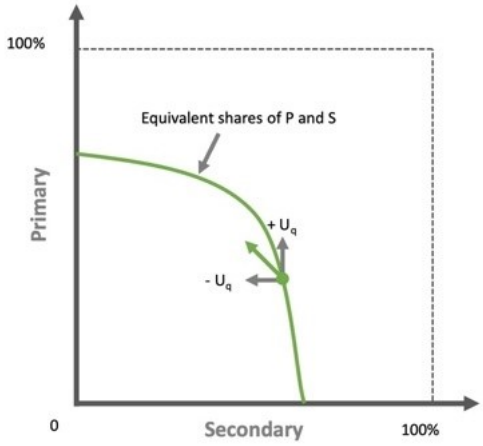


Figure 9: Forced unbalancing

This forcibly pushes the control towards the preferred axis: there's no guarantee that this will not introduce a temperature oscillation (it does, usually) because it considers P and S equally effective, and this is almost always wrong.
But as soon as the control is pushed out of neutral zone, the forced unbalancing stops, the PID regains control and drives the system safely to the newly preferred configuration.

D. *Total and partial free cooling modes*

Looking at the newly defined 2D control space, if we look at the axes as Water and Air instead of Primary and Secondary, we can recognize the operating modes that the design team initially expected to use:

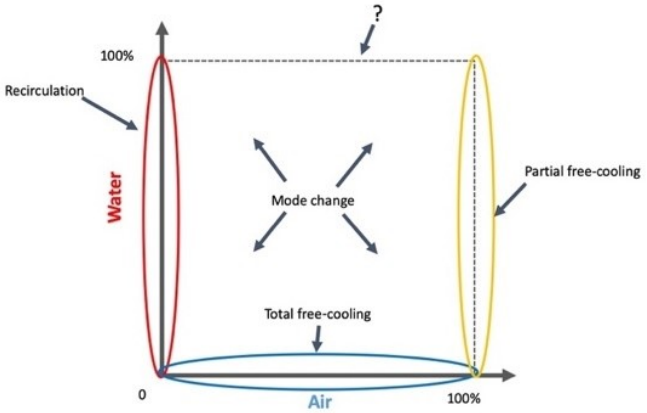


Figure 10: Operating modes in 2D control space

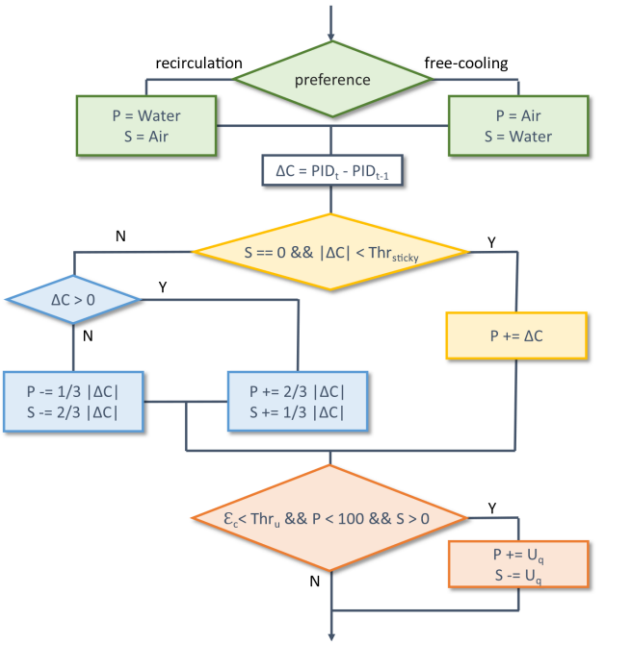
Whenever we change *source preference*, we just exchange the axes but the modes remain.
Looking at Figure 9 we can also understand what a "mode change" is in the newly defined algorithm: walking the green

line from one axis to the other. The line is undefined in advance, but the PID will "discover" it while walking.
In summary, the algorithm allows PID control to:

1. Provide continuous, smooth control, and reliably maintain setpoint during changes of *preference* between water and air
2. A change of preference may happen anytime, even during a "mode change"
3. Provides seamless backup during emergency conditions

Should the chilled water provisioning system fail, the best strategy would be to use air. It is better to open and get 35C than stay closed and let the data room temperature rise indefinitely. We could then consider that air is the backup of water. And, conversely, that water is the backup of air
The «partial free-cooling» is therefore the result of water backing up air when air would be preferred but is unusable.
The present algorithm seamlessly implements both water-as-air-backup and air-as-water-backup.

VI. ALGORITHM FLOWCHART



ε_c	control error (degrees from setpoint)
ΔC	differential PID output
PID_t	PID output at current timestep
PID_{t-1}	PID output at previous timestep
P	primary cooling source
S	secondary cooling source
Thr_{sticky}	stickyness threshold
Thr_u	unbalancing threshold
U_q	unbalancing quantum

Apply preference

Stickyness

Main control

Unbalancing

VII. MODE CHANGE EXAMPLE

Figure 11 demonstrates a real case of mode change, from recirculation to free-cooling. The system initially is in neutral zone, with 10% water usage and 0% air usage.
At 18:10 the preference change happened; being in neutral zone and no longer on the primary axis, the forced unbalancing kicks in and starts opening air and closing water.

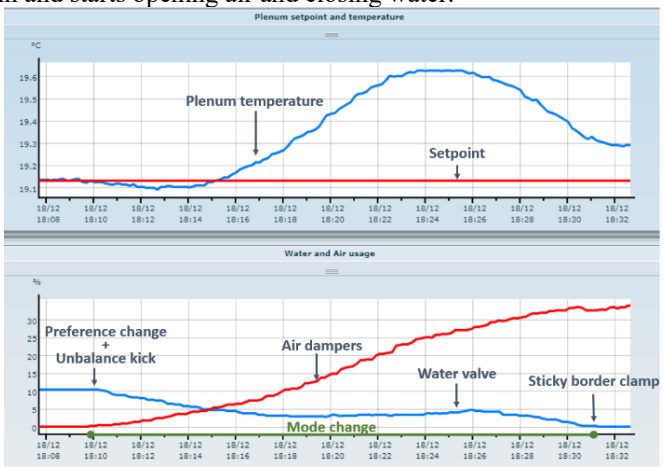


Figure 11: Mode change example

Around 18:14 the error introduced by the unbalancing (-.05C) moved the PID out of neutral zone: the unbalancing stops, and the PID begins applying the thirds rule with preference for air.
At 18:32 the water goes to 0% and, with cooling demand below the stickiness threshold, the sticky rule applies and clamps the system to using air only. The mode change can be considered complete.

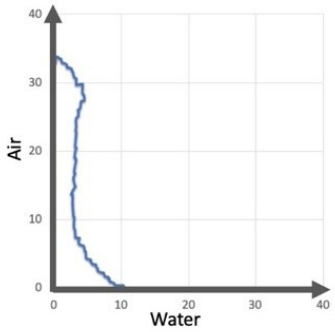


Figure 12: Discovered path of equivalent shares

Figure 12 shows the "equivalence path" walked by the PID, where we understand that *in that specific moment* 10% water was equivalent to 35% air.
It took about 20 minutes, with a maximum temperature error of 0.5

VIII. BEYOND AIR COOLING

The present algorithm has been developed to handle a case of multi-source cooling in an air-cooled datacenter.
But it could be easily adapted to many other cases, whenever different cooling sources are available and there is the need for strict temperature control while maintaining the freedom of

switching from one source to the other, or running them both together with a *preferred* one.

For example, a direct-watercooled HPC datacenter equipped with cooling towers on the chiller condenser circuit, may choose to use only the towers when possible, while letting the chiller contribute if needed.

It is also possible to extend the system to more than 2 sources: one just needs to assign to each source a preference value, and define a new "thirds rule" which distributes ΔC between the sources proportionally to the preferences.

IX. REFERENCES

[1] Bourrassa, N. et al, Operational Data Analytics: Optimizing the National Energy Research Scientific Computing Center Cooling Systems. ICPP Proceedings EE HPC SOP Workshop, Kyoto 2019

[2] Barclay, Craig "Maximizing Mission Capability by Minimizing Data Center Capacity", 2016, United States Department of Energy Energy Exchange Conference. Rhode Island

[3] Martinez, David J. " Integration of Building Controls and HPC Environments ", 2010, Energy Efficient High Performance Computing Working Group Liquid Cooling Controls Team Case Study

[4] Mingzhong, L. et al PID-Based Sliding Mode Controller for Nonlinear Processes https://pubs.acs.org/doi/abs/10.1021/ie990715e

[5] Sliding mode control: https://en.wikipedia.org/wiki/Sliding_mode_control

[6] Bortot, L. et al, Data centers are a software development challenge . Poster: ICPP Proceedings: Kyoto 2019 https://www.hpcs.cs.tsukuba.ac.jp/icpp2019/program/posters/index.html

[7] Bortot, L. et al, Data Centers are a software development challenge: https://eehpcwg.llnl.gov/assets/04_oda_paper_v2.pdf

[8] Martinez,D., SNL, Power Cooling and Design (Skybridge) Webinar: https://eehpcwg.llnl.gov/assets/100715_04_david_martinez_sky_bridge.pdf

[9] PLC: https://en.wikipedia.org/wiki/Programmable_logic_controller

[10] Coles, H. et al, "Hot" for Warm Water Cooling, SC'11, Nov.12-18, 2011 Seattle WA, USA https://eehpcwg.llnl.gov/pubs.html