Revised Paper for Journal of Process Control, Oct. 2026

Split-parallel control
- a little known control structure

Krister Forsman[*†], Mohammed Adlouni[*], Sigurd Skogestad[†]

**Perstorp Specialty Chemicals, 284 80 Perstorp, Sweden*

†Norwegian University of Science and Technology (NTNU), 7491, Trondheim, Norway

Abstract: We describe a control structure that is commonly used in the process industry, e.g. chemical and petrochemical industries, for switching between manipulated variables (MVs), but which has received little attention in academia. It has two controllers, typically PID-controllers, that control the same process value (y) but have different manipulated variables (u_i) and different setpoints (r_i). The scheme is sometimes called "separate controllers with different setpoints", but we suggest that a better name is "split-parallel control" (SPC), since the two controllers are placed in parallel in the block diagram, but the active control action is split between the two controllers, similar to in split-range control (SRC). SPC is an alternative to SRC, but it has some advantages compared to SRC, including the possibility to have different PID tunings for each MV. We also state some yet unresolved questions regarding the SPC structure, especially in regard to stability. Split-parallel control (SPC) uses setpoint separation to perform the switching, which is an advantage in some cases, for example, for bidirectional inventory control. This setpoint separation also allows for optimizing a cost criterion, when applicable.

Keywords: Process control, multivariable systems, control structures, control architecture, decentralized control, optimizing control, split-range control, MV-MV-switching, active constraint switching.

1. INTRODUCTION

Classical advanced control structures (architectures) are described in textbooks, e.g. in Marlin (2000), Smith and Corripio (2006), Skogestad and Postlethwaite (2005), as well in the recent review by Skogestad (2023). Some structures can be seen as "performance boosters" compared to PI-control, while others handle truly multivariable or non-linear control problems.

The processes that we study here have one controlled variable (CV; y), and (for simplicity) two manipulated variables (MVs; u_1, u_2). This means that we have an extra degree of freedom, which here is used to specify the sequence in which the MVs are used, that is, we want to use one MV at a time, in a predefined order, for example, based on using the cheapest MV first. A classical way of solving this problem is so-called split-range control (SRC), as described in Reyes-Lúa et al (2019). This is a case of what is known as MV-MV switching and two less known alternatives are "valve position control" and "separate controllers with different setpoints" (Skogestad, 2023).

In this paper, we focus on the last control architecture, which is commonly used in industry, but not much described in the literature. This structure has previously been called "separate controllers with different setpoints" (Forsman, 2005; King, 2011; Skogestad, 2023), but we suggest that it instead be called "split-parallel control" (SPC) because it is a parallel control structure which splits the control action and is an alternative to split-range control (SRC). Compared to SRC, the SPC structure has the advantage of being simpler to implement and allowing for separate controllers for each MV.

One potential disadvantage with SPC is that the switching is based on setpoint separation, which means the CV (y) must deviate from the original setpoint when the MV switching (between u_1 and u_2) occurs. In many cases, this is not a problem, or it may even be an advantage, for example, for the application to bidirectional inventory control (Shinskey, 1981; Zotică et al, 2022). In other cases, if we want convergence to a single setpoint, then it is possible to add an outer cascade loop to SPC as discussed in Section 7.

2. SPLIT PARALLEL CONTROL (SPC)

A block diagram for SPC is shown in Figure 1. The two controllers should have different setpoints (SP; r_1, r_2), but use the same process variable (PV)[1] , denoted y in the diagram.

In the block diagram, the process is represented by a single block P with two process inputs (u_1, u_2) which are equal to the controller outputs. We could equally well have written it as two processes with individual inputs, and the PV (y) as the sum of the outputs of those processes (assuming the system is linear and using the Laplace s-domain):

$$y = P_1(s)\, u_1 + P_2(s)\, u_2 + d$$

Here, d represents the effect of disturbances on y. If both controllers in Figure 1 have integral action, e.g. are PI-controllers, then with both controllers active the system would be internally unstable in a linear framework, as analyzed below. A simple way of understanding this is that, with both u_1 and u_2 being free to choose, there are infinitely many combinations of u_1 and u_2 that give the same y at steady state. However, in practical applications there are always controller output saturations (e.g., valve positions), and thus u_i are limited to the range 0% to 100%, or there are overrides from other controllers that limit the allowable range for u_i. The internal instability is actually an advantage as it drives one of the MVs towards its constraint limit (0% or 100%) and as we will see below this happens faster when the setpoint difference $|r_1\text{-}r_2|$ is large. This means that the two controllers in Figure 1 will be operated sequentially (except for shorter transition periods). That is, only one u_i is active at a time, and this makes the nonlinear system, with the saturations, stable.

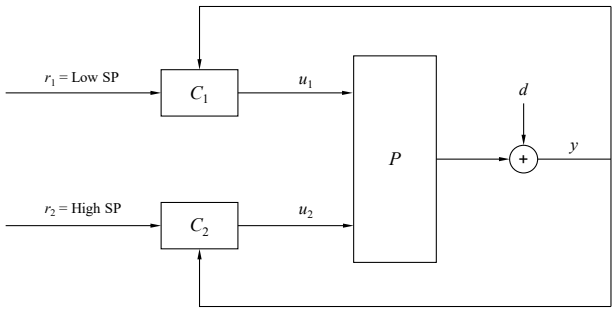

Figure 1. Block diagram for split-parallel control (SPC). Except during transitions, only one of the two controllers C_1 and C_2 is active. In this case r_1 is lower than r_2, but obviously it could be the other way around.

Thus, during normal operations, one of the controllers will be inactive, with u_1 or u_2 not being possible to manipulate, either because one of the MVs (u_1 or u_2) is at its maximum or minimum saturation limit, or because an override selector has made another controller take over one of the MVs. Obviously, anti-windup functionality needs to be included for the integral action of the controllers, and it plays an important role in how well the structure works.

This SPC structure is not explained or analyzed in much detail in the literature. It is briefly mentioned e.g. in King (2011), Forsman (2005, 2016), and also in Reyes-Lúa et al. (2018, 2019,2020) and Skogestad (2023). In Jagtap et al. (2013) it appears as a part of override control without being devoted special attention. Shinskey (1981) describes it in the context of bidirectional inventory control where the difference in setpoints is used to take advantage of the buffer capacity. The economic advantage of using different setpoints for heating a residence is discussed by Reyes-Lua and Skogestad (2019).

In Balchen and Mumme (1988) there is a mention of "parallel control", which has similarities to SPC, but the two controllers have the same setpoint. To avoid internal instability, only one controller has integral action. Here, the objective of the extra MV u_2 is to improve the dynamic response and both controllers are always active, so there is no split between the manipulated variables.

3. APPLICATION EXAMPLES

Example: Steam header pressure.

In this application there are two manipulated valves on individual pipes connected to a high-pressure steam header. One of the valves ($u_{2=}$ v_2,) is a vent valve which allows steam to be vented to the atmosphere. The other ($u_{1=}$ v_1,) sends high pressure steam through a turbine to the lower pressure header. The specifications for the control solution are that the high pressure header should normally be controlled to a setpoint of r_1=50 bar with the vent valve completely closed. The vent valve should only open when there is an excess pressure, that is, when the pressure reaches r_2=54 bar.

Figure 2 shows a split-parallel control architecture that satisfies the specifications. PC2 in the figure is the "rescue controller" that is only active when venting is needed. It has a higher setpoint than PC1, which is the controller maintaining the pressure in the header when the plant is running normally. Most of the time we wish the output of PC2 (u_2) to be zero, since venting is not desired. In this case PC2 could be a P-controller without integral action.

[1] In this paper, the terms PV (process variable) and CV (controlled variable) are used interchangeably, both referring to the process output y in Figure 1.

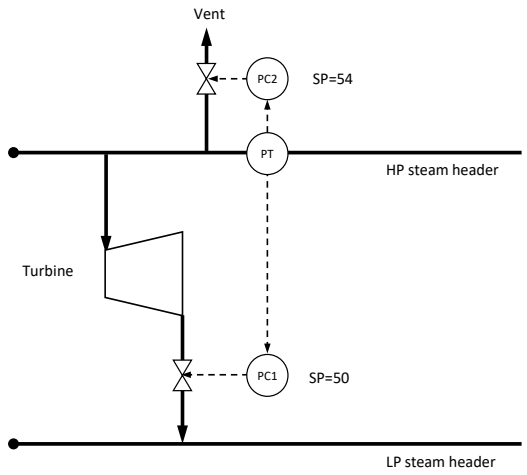

Figure 2. Split-parallel control for pressure in steam header.

Example: Bi-directional control.

Consider a chain of buffers (liquid inventories) connected serially by pipes. This is a simplistic representation of a plant with "linear" topology. If we wish to maximize the throughput of the plant, then the location of the master flow (throughput manipulator) should be at the location of the bottleneck. Using the control structure shown in Figure 3, we get automatic reconfiguration of level control – flow control pairing. Details are described in Zotică et al. (2022).

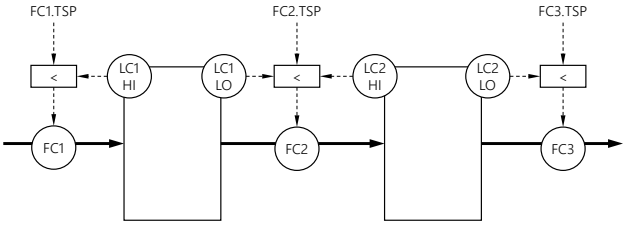

Figure 3. Bidirectional inventory control (Shinskey, 1981). Each tank has two level controllers with different setpoints (HI or LO), and the three min-selectors determine which one is active.. Reminder TO Krister: CHANGE < to MIN!

This is an example of a combination of split parallel controllers to switch between the MVs, and minimum selectors to switch the pairing of MVs and CVs. In this case, it is not important that the level (y) is at fixed setpoint, and each tank has a level controller with high setpoint (LC1HI, LC2HI) and a level controller with low setpoint (LC1LO, LC2LO). Compared to using SRC for the MV switching, the setpoint separation in SPC maximizes the throughput further by providing a "grace period" that delays the effect of a new bottleneck.

Example: House heating

In this application we have two sources of heating in a house, hot water (u_2) and more expensive electric heating (u_1). With split range control (SRC), we have one controller C and it is set up such that it first manipulates hot water (u_2) to control the temperature (y), and when it saturates, at u_2=100% (because it's a cold day), the controller switches to using electricity (u_1). With split parallel control (SPC), we have two controllers C_1 and C_2 with different setpoints, say r_2=23 °C for hot water and r_1=22 °C for electricity. If it is not so cold, we only use hot water (u_2), so u_1=*0* and the temperature is 23 °C. However, on a cold day, u_2 (hot water) will saturate at 100% and controller C_2 is not able to maintain the temperature at the setpoint r_2=23 °C. The temperature y will start dropping and when it approaches 22 °C, controller C_1 will activate and use the electric heat u_1 to keep y at the lower setpoint r_1=22°C.

Example: Catalytic incineration

In this application we consider the catalytic combustion of volatile organic compounds in exhaust gases. See Figure 4.

Here the controlled variable y is the temperature of the reactor feed. MV1 is the power of the electric heater before the reactor. Before the electric heater there is a heat exchanger that uses reactor exhaust gases to preheat the gas coming into the electric heater; and MV2 is the bypass of the heat exchanger. The optimization consists in using the electric heater (MV1) only when necessary, which is when the bypass saturates at MV2=0. This is achieved with SPC by having a higher setpoint for TC01B (MV1) than for TC01A (MV2). Note that the desired value of MV2 is determined by adjusting two valves – one on the bypass and one on the flow through the heat exchanger. Ideally, the second valve should be fully open. The optimal balance is achieved using split range control (SRC). In summary, both SPC and SRC are used for this application.

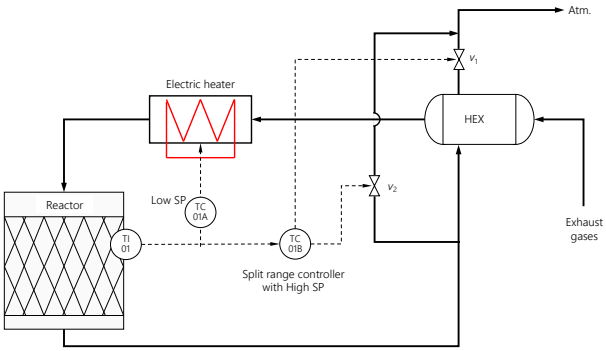

Figure 4: SPC for a catalytic incineration unit.

Example: Level with two feed streams

Consider a buffer between two plants, as shown in Figure 5. There are two feed streams, containing the same material but with different temperatures. The feed from a storage tank is cold, whereas the feed taken directly from the plant upstream is hot. We should avoid using the cold feed to control the level, if possible: that causes energy losses, since the feed to the downstream plant (Plant 2) would have to be heated.

Using SPC we get two level controllers: LC1 (cold liquid) having a low setpoint and LC2 (hot liquid) having a high SP. In this way there is a "grace period" (see Figure 8) and the cold stream will not be used unless necessary. The benefit will be larger with a large setpoint separation.

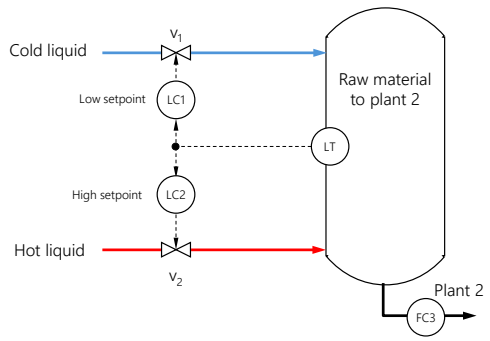

Figure 5: SPC for the level in a buffer tank.

4. THEORETICAL ANALYSIS OF SPC

From the above applications, the reader probably has an intuitive understanding of how split parallel control works by switching to the other MV (say u_2) if the current MV (say u_1) goes into saturation so that we "lose control" of the PV (y). That is, we switch between the following two regions:

- **Region 1 (partly constrained)**: u_1 controls y (at r_1) and $u_2=u_{2lim}$
- **Region 2 (partly constrained):** u_2 controls y (at r_2) and $u_1=u_{1lim}$

Here u_{1lim} and u_{2lim} denote thw constraints relevant for the switching. Depending on the case, it may be a minimum (say, 0%) or maximum constraint (say, 100%).

Note that a "direct" switch between regions 1 and 2 is unlikely with split-parallel control. For example, a direct switch from region 2 to region 1, requires that u_2 reaches saturation (u_{2lim}) exactly at the same time as controller C_1 makes u_1 go out of saturation (u_{1lim}). In practice, we will temporarily pass through either region 3 (fully constrained) or region 4 (fully unconstrained):

- **Region 3 (fully constrained transition region):** $u_1=u_{1lim}$, $u_2=u_{2lim}$ and y is "floating" (as given by the open-loop process dynamics and disturbances). Region 3 is associated with a "grace period" (delayed switch or no switch from the original MV constraint).
- **Region 4 (fully unconstrained transition region)**: Both u_1 and u_2 (attempt to) control y (at r_1 and r_2, respectively). Region 4 is associated with a "boosted" response followed by a possible "limbo" period.

A transition through region 3 is probably what most readers expect. For example, assume we are in region 2 (where u_2 is used to control y and $u_1=u_{1lim}$) and there is a disturbance so u_2 reaches u_{2lim} and we lose control of y. We then enter region 3 where both constraints are active. The PV y will then drift "naturally" (open-loop). At some point, y may have changed sufficiently towards r_1 to activate u_1 (and move to region 1) or the disturbance may have disappeared so that we return to region 2. Note that a transition through region 3 gives a "grace period" before the controller in the other region is activated.

A transition through region 4 is also possible, especially if the setpoint separation $|r_1-r_2|$ is small. For example, assume we are in region 2 (where u_2 controls y at r_2) and there is a disturbance which moves y towards r_1 (or y may even exceed r_1) such that controller C_1 activates u_1. Thus, both u_1 and u_2 are trying to control y at the same time, but at different setpoints. Initially, as we enter region 4, there is a "boosted" response because both controllers are active. This may be followed by a "limbo" period where the output (y) stays close to an "intermediate" steady state (between r_1 and r_2), but the two controllers are competing, resulting in slowly drifting MVs (u_1 and u_2). Eventually, one of the MVs (u_1 or u_2) will reach saturation and we enter region 1 or 2. Note that we say that it *may* enter a limbo period in region 4, because if the "new" saturating MV (which will end region 4) is close to its limiting constraint as we enter region 4, we may have an initial "boost" but never enter the limbo period. The limbo period in region 4 with slowly drifting MVs is associated with internal instability as discussed in the following.

For now, we will assume that both controllers have integral action. We consider the case where MV saturation (0% - 100%) is the reason for switching and we assume that the two controllers are tuned so that the entire system is stable, relying on the MV saturation.

Consider the simulation in Figure 6 where the initial values are chosen so that the output y seems to be at steady state, with y constant at a value (y=51) between r_1 and r_2. However, this is an "intermediate" or "limbo" state, because both controllers are active (Region 4) and they are competing to drive the system to their respective setpoints of r_1=47 and r_2=55. As seen from the simulation, the two MVs (u_1 and u_2) are not constant, but drifting towards their constraint limits. At about time t=8, controller 1 (C_1) saturates at u_1=0%, and the system enters Region 2 where controller 2 (C_2) drives the PV (y) from its initial intermediate state (y=51) to its setpoint (r_2=55).

In this simulation both processes P_1 and P_2 were first-order processes with deadtime, having the same dynamics. The controllers were PI controllers with the same tuning parameters (see Appendix for details).

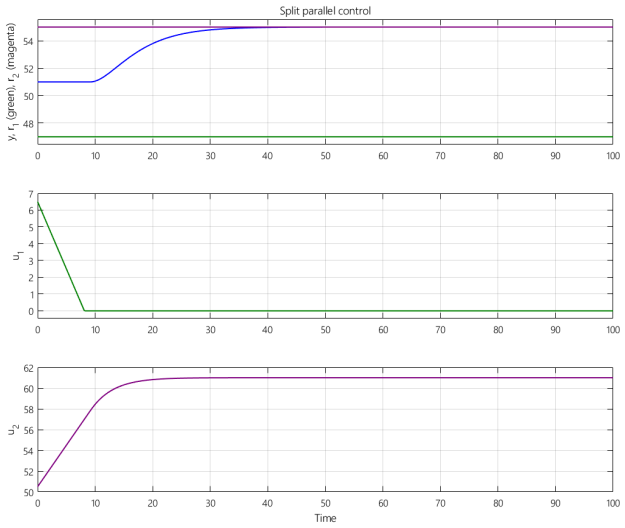

Figure 6:. Transient "limbo" behavior of an SPC system (region 4). Top: PV (y) and the two setpoints (r_1=*47,* r_2=*55*). Middle: Controller 1 output (u_1). Bottom: Controller 2 output (u_2).

The duration of the intermediate or "limbo" state - where the PV (y) is constant or approaching a constant value (between the setpoints r_1 and r_2), but the system is not in steady state because u_1 and u_2 are drifting - depends on several factors, including:

- Controller tuning (including anti-windup)
- Process dynamics
- Setpoint separation

Figure 7 illustrates that for the same initial condition and the same controller tuning, a smaller setpoint difference results in a slower convergence to steady state, i.e. a longer limbo period. In this simulation r_1=50.5 and r_2=51.5. Stationarity is achieved around t=75.

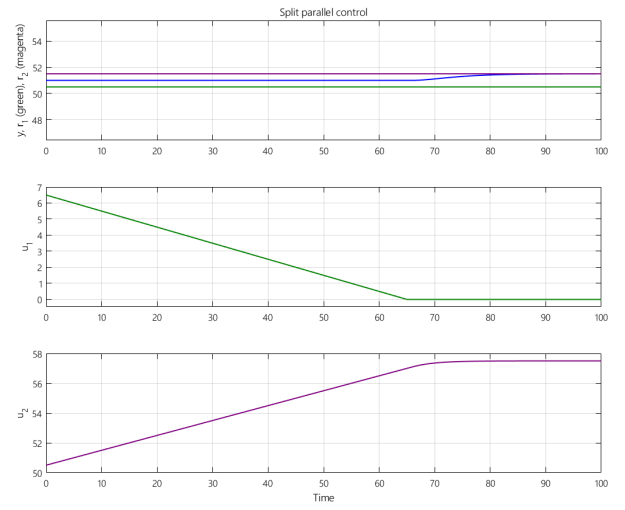

Figure 7. Transient behavior of an SPC system (region 4) with a long limbo period. Same processes and controllers as in Figure 6, but with smaller setpoint separation (r_1=*50.5,* r_2=*51.5*).

To explain the limbo period, assume that both parallel controllers are active so that the system is operating in the linear region 4 (with no constraints) with

$$u_1 = C_1(s)(r_1 - y), \quad u_2 = C_2(s)(r_2 - y)$$

where s denotes the Laplace variable. The closed-loop response of the linear system in Figure 1 is then (region 4)

$$y = \frac{1}{1+L_1+L_2}(L_1 r_1 + L_2 r_2 + d) \qquad (1)$$

where d represents the effect of a disturbance on y. Here L_1=P_1C_1 and L_2=P_2C_2.

Assume both controllers C_1 and C_2 have integral action. Then at steady-state, which corresponds to s=0 in the frequency domain, both L_1 and L_2 approach infinity because of the integrators in the controllers, and (1) gives that the system approaches the following intermediate "limbo" state:

$$y = \alpha r_1 + (1-\alpha) r_2 \;; \qquad \alpha = \frac{k_1 K_{I1}}{k_1 K_{I1} + k_2 K_{I2}} \qquad (2)$$

Here, k_i denotes the steady-state process gains and K_{Ii} the integral gain in the controllers. Note that α is between 0 and 1 so y approaches a value between the setpoints r_1 and r_2. Also note that the disturbance d has no effect on y at steady state with integral action in the controllers.

We note that the value for y in (2) (which the system may approach during the limbo period, but which is not truly a steady state because of the internal instability discussed below) is parameter-dependent in that it depends on the process gains and the tuning of the controllers. Say the ratio of the loop gains at low frequency $\frac{k_1 K_{I1}}{k_2 K_{I2}}$ is 1 (equal loops). Then, (2) gives that y during the limbo period approaches y = 0.5 r_1 + 0.5 r_2. If the ratio is 2 (higher gain in loop 1), we get y=0.67 r_1 + 0.33 r_2.

To see the internal instability, we need to consider the inputs. The closed-loop response from the setpoints r_i and disturbances d to u_1 and u_2 is given by (when operating in the linear region 4 with no saturation):

$$u_1 = \frac{C_1 r_1}{1+L_1+L_2} + \frac{C_1 C_2 P_2 (r_1 - r_2)}{1+L_1+L_2} - \frac{C_1 d}{1+L_1+L_2} \quad (3a)$$

$$u_2 = \frac{C_2 r_2}{1+L_1+L_2} - \frac{C_1 C_2 P_1 (r_1 - r_2)}{1+L_1+L_2} - \frac{C_2 d}{1+L_1+L_2} \quad (3b)$$

The internal instability for u_1 and u_2 appears in the middle terms in equation (3) where there is an "extra" integrator coming from the term C_1C_2 – both C_1 and C_2 have integral action - which is not stabilized by the feedback as given by the term $1/(1+L_1+L_2)$.

The internal instability results in both inputs having a constant integrating mode in periods when both parallel controllers are active. This appears as a constant drift towards the switching constraint for u_1 and u_2 – as seen during the initial limbo period in Figures 2 and 3 – which eventually drives one of the inputs towards its switching constraint limit (0% for u_1 or 100% for u_2). Which constraint is encountered depends on the value of the disturbance d.

During the limbo period, which corresponds to frequency 0 in the frequency domain, the gain of the two drifting terms for the inputs in (3) is proportional to

$$\frac{K_{I1}K_{I2}}{K_{I1}k_1+K_{I2}k_2}|r_1 - r_2|. \quad (4)$$

From (4) it follows that the drift towards the constraints will be faster – and shorten the limbo period - when the setpoint separation $|r_1\text{-}r_2|$ is large. This is confirmed in Figures 2 and 3, where a reduction in the setpoint separation by a factor 8 increased the limbo period by the same factor.

Note that (1), (2) and (3) assume that we are in the "linear unconstrained region" 4 where both controllers C_1 and C_2 are active. As soon as either u_1 or u_2 reaches a constraint or is taken over by another controller (e.g. through a MIN-selector), eqs. (1), (2) and (3) no longer apply.

5. COMPARISON TO SPLIT-RANGE CONTROL

In general, there are three alternative structures for MV switching (Skogestad, 2023), namely split-parallel control (SPC), split-range control (SRC) and split valve position control.

SRC is the oldest and most common structure. It always has a direct transition between regions 1 and 2. Compared to SRC, the less well-known SPC alternative studied in this paper, has some advantages:

- With SPC, you can tune the two controllers differently so that the parameters, like integral time, are fit for the subprocess manipulated with that controller. In SRC, one may indirectly achieve different controller gains for the two subprocesses by tuning the split location in the table, but it is not possible to have different integral times, unless you use gain-scheduling, which could in this case become complicated.

- When using SRC, the two valves are algebraically locked to each other. The user cannot easily choose to run the valves independently, whereas this is immediately obvious how to do with SPC.

- SPC may offer an economic advantage for cases where a large setpoint separation is acceptable. The reason is that the constraint transition is likely to go through the fully constrained region 3. This results in a "grace period" which may avoid or delay the need to switch to using a more expensive input.

  In the bidirectional control application example above, we could have used a single SRC for each level and combined those with selectors. However, the advantage of using SPC with a large setpoint separation is that the varying steady state buffer levels give a grace period (region 3) that delays the switching and enables production to be continued when there is a new bottleneck, as explained in Zotică et al (2022).

  Another advantage of the "grace period" for SPC in region 3, is that a disturbance of short duration does not necessarily cause a switch to the other MV. This happens if it is acceptable to have a large setpoint separation, like for the two examples with level control (Figures 3 and 5). This is illustrated in Figure 8, where a pulse disturbance enters at time t=20 and disappears at t=60. Because y is allowed to vary (at least in the range from r_1=20 to r_2=90), we avoid using the "expensive" input u_1 in spite of the fact that u_2 saturates at 100%. The alternative solution with split-range control (SRC), on the other hand, will not be optimal cost-wise, since it, when u_2 saturates at 100%, will use u_1=v_1 unnecessarily to keep y at a constant setpoint.

- If the setpoint separation is small, it is likely that the constraint transition for SPC goes through the unconstrained region 4, where both manipulated variables (u_1 and u_2) are used simultaneously. In terms of the output y, the response may be improved compared to SRC. This "boosted" response is illustrated in Figure 9 where the disturbance at time t=40 causes C_1 to be active for a short time. This speeds up the response for y to return to the desired setpoint range $[r_1,r_2]$, see eq. (2). This will not happen with conventional split-range control, which would only manipulate one MV at a time during transients.

The "boosted response" is also a potential disadvantage with SPC because stability problems may arise during transients when the two parallel controllers are active simultaneously as the total loop gain increases. However, in practice, this is believed to be more a theoretical rather than a practical problem (see Discussion Section 6.1).

As mentioned, a more important disadvantage of the SPC structure with a small setpoint separation is that we during an unconstrained transition (region 4) may enter a "limbo state" that delays the desired switching of the MVs. This can be observed also for the "boosted" response in Figure 9. We see that after the output y returns to the setpoint range $[r_1,r_2]$, the limbo effect causes u_2 to move slower towards its new desired steady state and the output y temporarily approaches the intermediate value of 51 between r_1 = 50 and r_2= 52.

The SPC architecture obviously has the potential drawback that the value of y - which setpoint (r_1 or r_2) will be reached at steady state - depends on the process state, e.g. disturbances. However, this can be addressed by adding another "master" controller on top of the other two, as explained in Discussion Section 6.2.

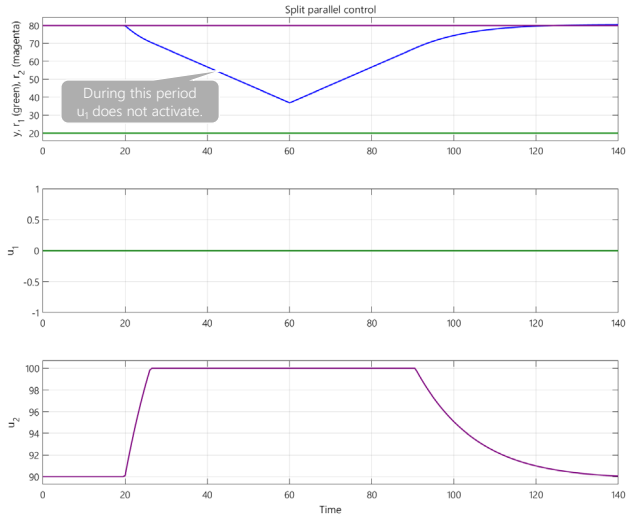

Figure 8. SPC simulation illustrating "grace period" (Region 3) for a temporary disturbance (pulse from t=20 to t=60) that keeps the output y within the desired setpoint range $[r_1,r_2]$ = [20,80]. The simulation is for an integrating process with a large setpoint separation. The application could be the buffer level control illustrated in Figure 5 where we want to avoid using the expensive cold feed u_1=v_1 whenever possible. The disturbance is a temporary increase in the flow to Plant 2.

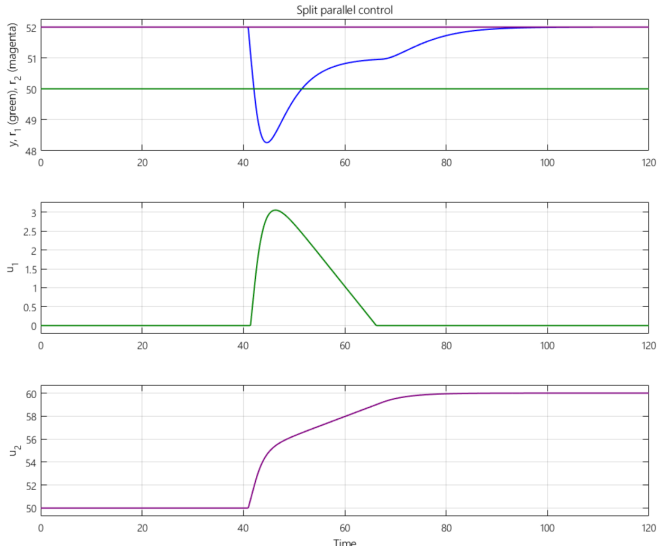

Figure 9: "Boosted" transient response by SPC (Region 4) when a large step disturbance at t=40 causes the output y to go outside the desired setpoint range $[r_1,r_2]$ and input u_1 is temporarily activated.

6. DISCUSSION

6.1 Stability When studying SPC, a question posing itself immediately is: What are the conditions on system dynamics and controller tuning for the two controllers not to start interacting in such a way that the entire system becomes unstable?

The internal instability of the linear system in region 4 was discussed following equation (3). It may result in what we referred to as a temporary limbo state, but eventually one of the inputs will saturate and the system enters region 1 or region 2, which can be analyzed using a conventional linear stability analysis, using, for example, the gain margin (GM) or the peak of the sensitivity function (M_s) as robustness measures.

A potential "nonlinear instability"[2] – which is a disadvantage of the boosted setpoint response in region 4 - may occur if the controllers are tuned too aggressively. Assume that the individual controllers are tuned so that the system is stable when only one controller is active (in the linear regions 1 or 2). The question is: Is it possible that the nonlinear system goes unstable when a disturbance causes a switch to region 4 where both controllers are active? Intuitively, the answer is yes if we use too aggressive control, but how aggressive does it have to be?

Imagine that we have a system with two identical loops (L_1=L_2). We then have the same gain margin GM1=GM2 for the individual loops (in regions 1 and 2). The gain margin for the linear system with both controllers active (region 4) will then be reduced by a factor 2 (GM=GM1/2). This follows because the loop transfer function in region 4 is $L = L_1+L_2 = 2L_1$, so the overall loop gain is multiplied by factor 2. Since a GM less than 1 is equivalent to linear instability, it follows that the intermediate "limbo" state in region 4 will be linearly unstable only if GM1=GM2 is less than 2, which requires fairly aggressive tuning of the individual loops. However, "linearly unstable in region 4" does not mean that the overall nonlinear system, with the saturation constraints, is "nonlinearly unstable". The reason is that when an input reaches a constraint, the unstable drifting stops and the system becomes linearly stable. We don't have a general theoretical analysis of when stability will be achieved, because stability analysis is difficult for constrained systems with switching. However, we find in simulations that we have to reduce GM1=GM2 below 2 (depending on the setpoint separation) before we encounter "nonlinear instability". Consider the following example;

$$P_1(s) = P_2(s) = \frac{\exp(-2s)}{5s+1}, C_1(s) = C_2(s) = K_c\frac{5s+1}{5s}$$

Figure 10 with K_c=3 shows that we have "nonlinear instability" with limit cycles where y varies between 30 and 90. However, note that the setpoint separation is small (r_1=50, r_2=51) and that K_c=3 is a very aggressive controller gain (with GM1=GM2=1.31 much less than 2). A linear stability analysis gives that the individual loops are unstable for $K_c > 3.93$, and thus the combined linear system in region 4 is unstable for $K_c > 3.93/2 = 1.96$. However, simulations show that we can increase K_c to 2.6 before we get the "nonlinear instability" observed in Figure 10. If we reduce K_c say is "unstable" - may not satisfy mathematical definitions of instability.

[2] We put «nonlinear instability» in quotes, because a (stable) limit cycle as seen in Figure 10 - which most people would

below 2.6, the oscillations in Figure 10 die out and the system approaches a stable steady state with y=r_2=51 and u_1=0%. ~~Another way to avoid "nonlinear instability" is to increase the setpoint separation. For the current example, simulations with r_1=50 and r_2=52, show that the controller gain K_c must be increased to about 4.5 before we encounter limit cycles as in Figure 10. On the other hand, if we decrease the setpoint separation towards zero (r_1=50, r_2=50), the simulations show that system develops a limit cycle for K_c>1.96 (as expected from the linear analysis)~~.

.

In summary, aggressive tuning can produce limit cycles during Region 4 operation (Figure 10). This behavior is less likely under typical industrial proportional-integral-derivative (PID) tunings, for which the individual-loop gain margins are often 3 or larger, as with the simple internal model control (SIMC) rule. If such oscillations occur, they can be mitigated by detuning one or both controllers or by increasing the setpoint separation.

Nevertheless, a more careful theoretical study of such switched systems would be useful to answer the overall "nonlinear" stability issues in greater detail.

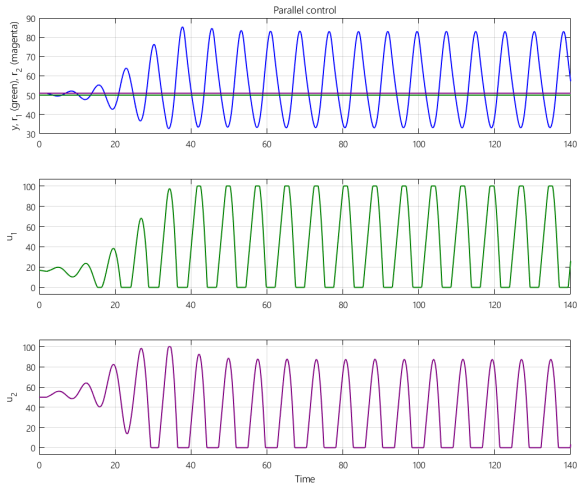

Figure 10. Simulation scenario of boosted response in region 4 with limit cycles ("nonlinear instability") caused by aggressively tuned individual controllers with GM=1.31 (K_c=3) and a small setpoint separation (r_1 = 50, r_2 = 51).

6.2 Cascade control for unique steady state

One disadvantage of SPC is that it requires a setpoint separation. Figure 11 shows an extension of the SPC scheme, where an extra controller C_0 with a master setpoint r_0 has been added (Skogestad, 2023). This slower "master controller" gives the setpoint r_1 and r_2 for C_1 and C_2, with a fixed bias Δr in between them. In this architecture the process value y will always converge to the setpoint r_0 in steady state (assuming stability).

As discussed above, a fixed setpoint r_0 is desirable in many applications, but there are exceptions, for example, if y is an inventory (level) which does not have a fixed setpoint, but should be bounded between the values r_1 and r_2.

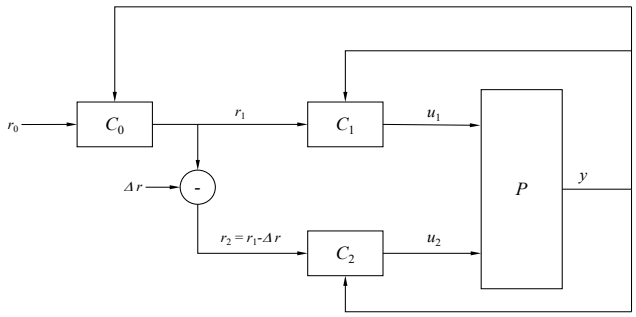

Figure 11. SPC extended with a slower master controller C_0(with I-action) which adjusts the individual setpoints r_1 and r_2 (with a fixed setpoint separation Δr) such that the process value y always approaches the setpoint r_0 at steady state.

7. CONCLUSIONS

Split-parallel control (SPC) is commonly used in industry for cases where MV-saturation makes it necessary to use two or more MVs (u_1, u_2) to control the CV (y) under all steady-state conditions. SPC is internally unstable when seen from a linear point of view (in transition region 4), but this is actually an advantage as it drives one of the MVs to its constraint limit.

Compared to the more well-known split-range control, SPC has the advantage that it can have different tunings for each MV. It requires the use of different setpoints (r_1, r_2) which may be an advantage or disadvantage, depending on the situation.

The traditional split-range control (SRC) results in a "direct" transition between the two constraint regions (denoted regions 1 and 2 in this paper). On the other hand, with SPC, the transition goes through either a fully constrained region 3 (with a "grace period" for the constraint switching) or an unconstrained region 4 (with an initial "boosted response" followed by a possible intermediate "limbo state"). A transition through region 3 is more likely with a large setpoint separation, whereas a transition through region 4 occurs with a small setpoint separation.

In reference to economic optimization, SPC with a large setpoint separation has the advantage of allowing a "grace period" for disturbances (region 3 transition), where the more costly MV is not used until really needed. In particular, for buffer tanks (Figures 3 and 5) it is usually not required (or even desired) to keep a constant level, so we may use a large setpoint separation. For example, in the simulation in Figure 8 the level is allowed to vary between a low level setpoint of 20% and a high level setpoint of 80%. In Figure 5, we alternate between manipulating two inflows, but a more common case is bidirectional inventory control (Figure 3), where the inflow is manipulated at the high inventory (level)

setpoint and the outflow is manipulated at the low inventory setpoint. This idea may be extended to many tanks (units) in series as illustrated in Figure 3. The "grace period" results in optimal use of the buffers (intermediate storage) when the production bottleneck moves (Zotica et al, 2022).

An application where it is may be optimal to have a setpoint separation is control of room temperature (Reyes-Lua and Skogestad, 2019). Here it is economically optimal to have a high temperature setpoint when cooling is needed (summer) and a low setpoint when heating is needed (winter). However, for acceptable comfort the setpoint separation should not be too large. The main potential disadvantage with the SPC solution is that with a small setpoint separation, SPC systems can end up in what we call a "limbo state", where neither controller output (MV) is at its saturation limit, which delays the switching to a more favourable MV. In most cases, the limbo state is undesirable, although the "boost" effect in region 4 may give better transient control of the CV (y). The possibility of encountering the limbo state is difficult to predict as it depends on many factors, including the process model, controller and disturbances. However, the following measures will help shorten or completely avoid the limbo period:

1. Increase the setpoint separation $|r_1\text{-}r_2|$.
2. Increase the integral gain K_I for the slowest loop (with the smallest value of $K_I k$). Since $K_I=K_c/\tau_I$ this may be done by increasing the controller gain K_c and/or decreasing the integral time τ_I.
3. Usie less anti-windup action for the integral part of the controller (e.g., increasing the tracking time τ_T)

Measures 1 and 2 follow from equation (4), which shows that the duration of the limbo period is inversely proportional to the magnitude of $\frac{K_{I1}K_{I2}}{K_{I1}k_1+K_{I2}k_2}|r_1 - r_2|$. Measure 3 follows because the limbo state is encountered when an input (u_1 or u_2) goes out of saturation and this is delayed when the integral part of the controller "winds up".

If the required setpoint separation to avoid the limbo period is larger than can be accepted at steady state, but it is acceptable to have a fairly large transient deviation from the setpoint r_0 during switching, then the outer cascade in Figure 11 may be a good solution. If none of these measures are sufficient, then one may consider using split range control.

Acknowledgements. Helpful comments from Hyuntae Kim are gratefully acknowledged.

REFERENCES

J.G. Balchen and K.I. Mumme: *Process Control: Structures and Applications*. Kluwer, 1988

K. Forsman: Implementation of advanced control in the process industry without the use of MPC. *DYCOPS conference, Trondheim, IFAC papers online*, 49, pp 514–519. 2016.

K. Forsman: *Reglerteknik för processindustrin*. Student-litteratur, 2005. (In Swedish)

R. Jagtap, A.S. Pathak and N. Kaistha: Economic Plantwide Control of the Ethyl Benzene Process. *AIChE Journal*, Vol. 59, No. 6, pp 1996-2014. June 2013.

M. King: *Process Control. A Practical Approach*. Wiley, 2011.

T.E. Marlin: *Process Control*. 2nd edition. McGraw Hill, 2000.

A. Reyes-Lúa and S. Skogestad: Multiple-Input Single-Output Control for Extending the Steady-State Operating Range—Use of Controllers with Different Setpoints. *Processes*, 7, 941, 2019.

A. Reyes-Lúa and S. Skogestad: Systematic Design of Active Constraint Switching Using Classical Advanced Control Structures. Ind. Eng. Chem. Res., 59, pp 2229-2241, 2020.

A. Reyes-Lúa, C. Zotică, and S. Skogestad: Optimal operation with changing active constraint regions using classical advanced control. *IFAC papers online from IFAC Adchem Symposium, China, 2018*

A. Reyes-Lúa, C. Zotică, K. Forsman and S. Skogestad: Systematic Design of Split-range Controllers. *IFAC papers online* from 12th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS. Florianópolis, Brazil. 2019.

F.G. Shinskey: *Controlling Multivariable Processes.* Instrument Society of America, 1981.

S. Skogestad and I. Postlethwaite: *Multivariable Feedback Control: Analysis and Design.* 2nd edition. Wiley, 2005.

S. Skogestad: Advanced control using decomposition and simple elements. *Annual Reviews in Control,* vol. 56, Article 100903 (44 pages). 2023

C.A. Smith and A. Corripio: *Principles and Practice of Automatic Process Control*. 3rd Edition. Wiley, 2006

C. Zotică, K. Forsman and S. Skogestad: A simple bidirectional inventory control structure with optimal use of intermediate storage. *Computers and Chemical Engineering*, 159, 107677. 2022.

Appendix. Simulation parameters

Figures 6, 7 and 9:

First-order processes with unit gain, time constant 5 [time units] and time delay 1:

$$P_1(s) = P_2(s) = \frac{\exp(-s)}{5s+1}$$

Where s denotes the Laplace variable. The PI-controllers have an integral time equal to the process time constant 5 and a controller gain 0.83 (tuned according to the SIMC rules with closed-loop time constant $\tau_c = 5$):

$$C_1(s) = C_2(s) = 0.83\frac{5s+1}{5s}$$

Anti-windup with tracking (e.g., Skogestad (2023)) is implemented on both controllers (see below). A common rule of thumb is to choose the tracking time constant equal to or somewhat smaller than the integral time (which is 5 in this case). In this case, we select the tracking time constants $\tau_{T1} = \tau_{T2} = 2$.

Figure 8:

Integrating process with PI-control (tuned according to the SIMC rules with τ_c =10):

$$P_1(s) = P_2(s) = \frac{0.1}{s}$$

$$C_1(s) = C_2(s) = \frac{40s + 1}{40s}$$

Anti-windup with tracking (e.g., Skogestad (2023)) is implemented on both controllers with tracking time constants $\tau_{T1} = \tau_{T2} = 20$.

Figure 10 ("nonlinear instability"):

This is similar to the case in Figures 6, 7 and 9, but with a larger process time delay of 2:

$$P_1(s) = P_2(s) = \frac{\exp(-2s)}{5s + 1}, C_1(s) = C_2(s) = K_c \frac{5s + 1}{5s}$$

and a much larger controller gain K_c=3. Anti-windup with tracking is implemented on both controllers with tracking time constants $\tau_{T1} = \tau_{T2} = 2$.

PI control using anti-windup with tracking

The proper implementation of anti-windup is critical when using split-parallel control. For controller C_i, the error is $e_i = r_i - y$, and the simulations use the following tracking-PI realization:

$u_i = \min\{100, \max\{0, w_i\}\}$

where the internal ("unsaturated") controller output w_i is the sum of a proportional and integral contribution:

$$w_i = K_{ci} e_i + I_i$$

$$\frac{dI_i}{dt} = \frac{K_{ci}}{\tau_{Ii}} e_i + \frac{u_i - w_i}{\tau_{Ti}}$$

When an MV u_i is fixed at its limit (0 or 100%), its controller is considered inactive in the operating sense used in this paper, although its internal PI calculation continues. Thus, w_i will continue changing until $\frac{dI_i}{dt} = 0$ and it reaches a steady-state value,

$$w_i = u_i + \frac{K_{ci}\tau_{Ti}}{\tau_{Ii}} e_i$$

where u_i is at its saturation limit (typically, 0 or 100%). Note that $e_i = r_i - y \neq 0$ for an inactive controller, so in this case w_i is not equal to u_i. The difference between w_i and u_i (sometimes called the "windup depth" or "saturation depth") is equal to $w_i - u_i = \frac{K_{ci}\tau_{Ti}}{\tau_{Ii}} e_i$ so it is smaller when the tracking time τ_{Ti} is small. The "windup depth" (which may be positive or negative) tells how far the internal controller output w_i has moved beyond its saturation limit. Some "windup depth" is desired to avoid that u_i unnecessary comes out of saturation (0 or 100%) in the presence of disturbances and noise. A common rule of thumb is to choose the tracking time constant τ_{Ti} equal to or somewhat smaller than the integral time τ_{Ii}. Note that with a large setpoint separation (resulting in a large value of e_i) the "windup depth" may become very large, but this is not a problem as it will be reduced as y approaches r_i which is when u_i is supposed to go out of saturation. In the simulations, the initial value of the integral states, $I_1(0)$ and $I_2(0)$, are set such that $u_i(0) = K_{ci}\big(r_i(0) - y(0)\big) + I_i(0)$. Consequently, two runs with different setpoints require different initial integral contributions. This is different from changing the setpoints during a run, when the integral states , $I_1(t)$ and $I_2(t)$, remain continuous.