Power Round

Names: _____

Team Name: _____

Instructions

1. Do not begin until instructed to by the proctor.
2. You will have 90 minutes to solve the problems during this round.
3. Your submission will be graded and assigned point values out of the total points possible per problem. Your total score will be the sum of the points you receive for each problem.
4. Submissions will be graded on correctness as well as clarity of proof. A proof with significant progress towards a solution may receive more credit than a correct answer with no justification.
5. **You may use the result of a previous problem in the proof of a later problem, even if you do not submit a correct solution to the referenced problem.** However, you may not use the result of a later problem in the proof of an earlier problem.
6. Please submit each part of each problem on a separate page. Write your team name, problem number, and page number clearly at the top of each page.
7. No calculators or electronic devices are allowed.
8. All submitted work must be the work of your own team. You may collaborate with your team members, but no one else.
9. When time is called, please put your pencil down and hold your paper in the air. **Do not continue to write.** If you continue writing, your score may be disqualified.
10. Do not discuss the problems with anyone outside of your team until all papers have been collected.
11. If you have a question or need to leave the room for any reason, please raise your hand quietly.
12. Good luck!

Acceptable Answers

1. Solutions should be written in proof format. All answers, reasoning, and deductions must be explained and justified, unless the problem explicitly asks for you to "compute". Problems asking you to "show", "prove", or "justify" **require proof!**
2. Proofs will be graded both on correctness as well as clarity of presentation.
3. Partial credit may be awarded for significant progress towards a solution.
4. Each problem must be written starting on a new, blank page. Two different problems should not be written on the same page.
5. At the top right corner of each page, please clearly print your team name, problem number, and page number.
6. Answers must be written legibly to receive credit. Ambiguous answers may be marked incorrect, even if one of the possible interpretations is correct.

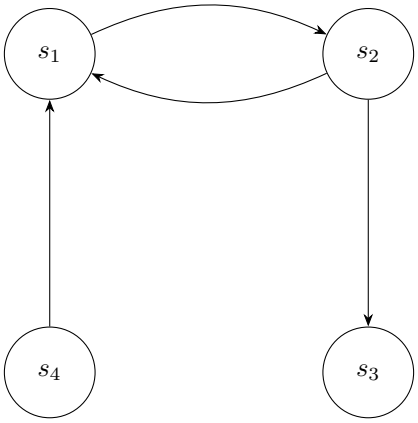
The Diamond Dilemma

Introduction

Definitions and Notation: A *one-player game* consists of a set of states $S = \{s_1, s_2, s_3, \dots\}$. For each state $s_n \in S$, there is a (potentially empty) set of states $S_n \subseteq S \setminus \{s_n\}$ that the player can move to from s_n.

We can represent a one-player game with a diagram that uses circles for states and arrows for the possible moves between them.

For example, a game with four states $S = \{s_1, s_2, s_3, s_4\}$, where the set of possible moves from s_1 is $S_1 = \{s_2\}$, the set of possible moves from s_2 is $S_2 = \{s_1, s_3\}$, the set of possible moves from s_3 is $S_3 = \{\}$, and the set of possible moves from s_4 is $S_4 = \{s_1\}$, can be drawn as follows:

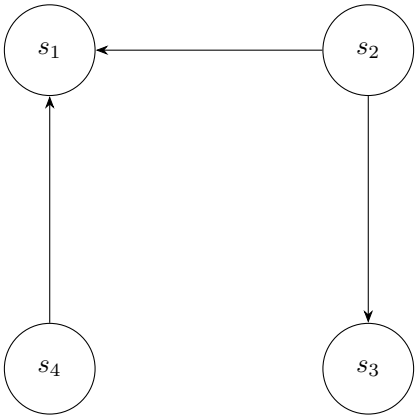


From now on, we will use these diagrams instead of the set notation used above to describe one-player games. We encourage you to do so as well in your proofs and examples.

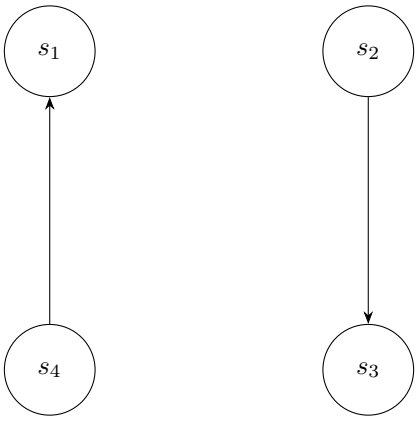
We say that a state s_n is *terminal* if $S_n = \{\}$. In other words, there are no moves that can be made from s_n. In the example above, s_3 is terminal, but s_1, s_2, and s_4 are not terminal.

We now define three properties that a one-player game can have:

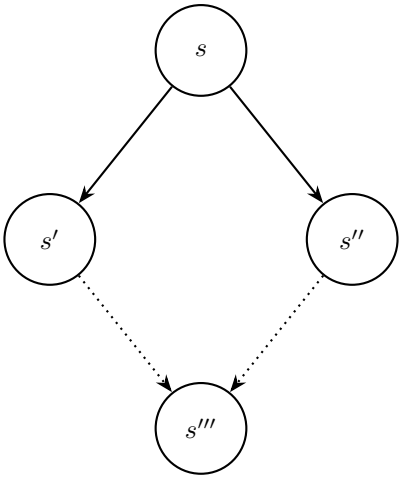
We say that a one-player game is *terminating* if there does not exist an infinite sequence of moves. The example above is *not* terminating because the player may move infinitely back and forth between s_1 and s_2. However, the following example below *is* terminating because no matter where the player starts, they will have to end up at either of the terminal states s_1 or s_3 in a finite number of moves:



We say that a terminating one-player game is *confluent* if choosing a starting state determines a unique terminal state that the player must end up reaching no matter what sequence of moves they choose. Note that this does not mean that *every* starting state leads to the *same* terminal state. The second example above is *not* confluent because starting from s_2, it is possible to reach either of the terminal states s_1 or s_3. However, the following example below *is* confluent because starting from s_1 or s_4 forces the terminal state to be s_1, and starting from s_2 or s_3 forces the terminal state to be s_3.



We say that a one-player game satisfies the *diamond condition* if whenever there is a state s such that there exists a choice of two states s' and s'' that the player can move to from s, then there also exists a fourth state s''' such that there exists a (possibly empty) sequence of moves from s' to s''' as well as a (possibly empty) sequence of moves from s'' to s'''. The diamond condition is illustrated below, using a dotted arrow to represent a sequence of moves.



Note that this property has to apply to *every* branch s, s', s'' within the one-player game in order for it to satisfy the diamond condition. Also, note that the diamond condition does *not* imply that *every* sequence of moves starting from s' or s'' must eventually reach s'''. It only specifies that such a sequence of moves exists.

Problem 1. The *Diamond Lemma* states that a terminating one-player game is confluent if and only if it satisfies the diamond condition. In order to build intuition for this lemma, we will first ask you to provide examples of one-player games.

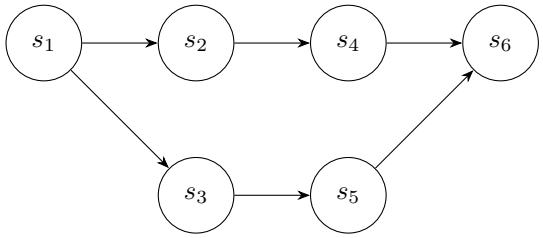
(a) (1 Point) Give an example of a one-player game with exactly 6 states that is terminating, confluent, and satisfies the diamond condition. Briefly justify each property.

(b) (1 Point) Give an example of a one-player game with exactly 6 states that is terminating but is neither confluent nor satisfies the diamond condition. Briefly justify each property.

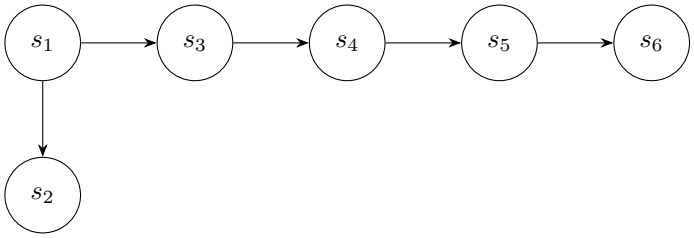
(c) (2 Points) Give an example of a one-player game with an infinite number of states that is terminating, confluent, and satisfies the diamond condition. Briefly justify each property.

Solution:

(a) Teams were given one point if they drew or described a one-player-game with exactly 6 states that was *terminating*, *confluent*, and followed the *diamond condition*, along with a brief explanation of how their game satisfied each condition. The game did not need a state that transitioned to more than one state to earn the point (envoking the diamond condition). Drawings with no transitions were also considered correct. Consider the example:

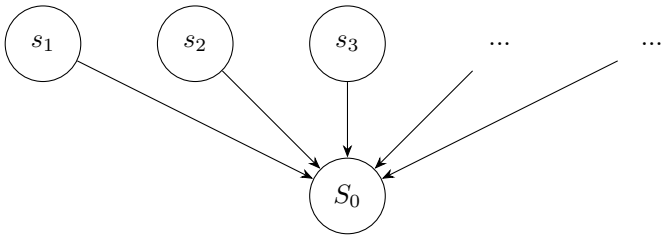


(b) Teams were given one point if they drew or described a one-player-game with exactly 6 states that was *terminating*, **not** *confluent*, and did **not** follow the *diamond condition*, along with a brief explanation of how their game satisfied or did not satisfy each condition. Consider this example:



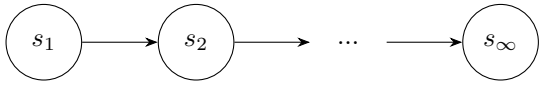
Most teams that did not earn this point drew one-player-games that **were** *confluent*.

(c) Teams were given 2 points if they drew or described a one-player-game with an infinite number states that was *terminating*, *confluent*, and followed the *diamond condition*, along with a brief explanation of how their game satisfied each condition. One main consideration was that there are a finite number of transitions. Consider this example which has at most 1 transition:



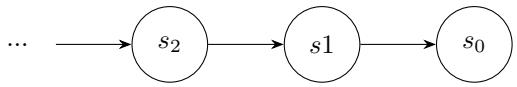
1 point was taken off if the explanation was incorrect or not coherent.

Most teams that did not earn these 2 points drew one-player-games suggesting an infinitely long path to a state s_∞ (sometimes labeled s_n). This means that their game is not *terminating* as there exists an infinite sequence of moves. Teams that suggested this were given 0 points. Examples of 0-point solutions are below.



$S_i = \{S_{i+1}\}$ for $i \geq 0$

However, teams were given 2 points if they suggested an end S_0, defined transitions working backward, and had no state going into the beginning of the sequence (for example $S_{n+1} = \{S_n\}$ for $n \geq 1$). An example is given below.



$S_{i+1} = \{S_i\}$ for $i \geq 0$

Problem 2. We will now prove the Diamond Lemma.

(a) (1 Point) Prove that if a terminating one-player game is confluent, then it satisfies the diamond condition.

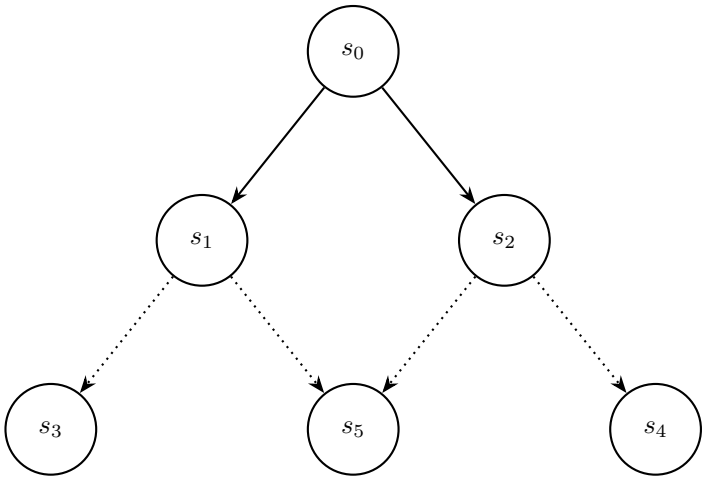
(b) (6 Points) Prove that if a terminating one-player game satisfies the diamond condition, then it is confluent.

Solution:

(a) Suppose that S is a terminating confluent one-player game. Let $s_1 \to s_2, s_3$ be an arbitrary branch in S. Since S is terminating and confluent, there exists a terminal state s_4 such that every sequence of moves starting at s_1 ends at s_4. Since s_2 and s_3 are both children of s_1, every sequence of moves starting at either s_2 or s_3 also ends at s_4. Specifically, such sequences exist because otherwise, a sequence starting at s_2 or s_3 would have to terminate at some state other than s_4, a contradiction.

(b) One possible proof strategy is to use induction. We will use contradiction instead. Suppose that S is a terminating one-player game that satisfies the diamond condition. Suppose for the sake of contradiction that S is non-confluent. Let us call states that break the confluent condition *bad* (i.e. states that can reach two or more different terminal states). Let us call all other states *good* (i.e. states that can reach exactly one terminal state). Since S is non-confluent, there exists at least one bad state in S. Starting at this bad state, make moves to other bad states until it is impossible to do so (until all children states are good). This process always terminates because S is terminating and all terminal states are by definition good states. Call this final bad state s_0. Since s_0 is bad, there exists two different terminal states s_3 and s_4 that are reachable from s_0. Let s_1 and s_2 be the direct children of s_0 on the paths to s_3 and s_4, respectively. Since s_0 has only good children, s_1 and s_2 are good states. By the diamond condition, there exists a terminal state s_5 that is reachable from both s_1 and s_2. Since s_1 is good, it can reach exactly 1 terminal state, but s_1 can reach both s_3 and s_5, so s_3 must be the same state as s_5. Since s_2 is good, it can reach exactly 1 terminal state, but s_2 can reach both s_4 and s_5, so s_3 must be the same state

as s_4. Therefore, s_3 must be the same state as s_4. But this is a contradiction since s_3 and s_4 were two different terminal states. Therefore, S must be confluent.



Problem 3. We will now prove the validity of *Bubble Sort*, a sorting algorithm used to sort a list of numbers. The Bubble Sort one-player game is defined such that a state is any finite ordered sequence of unique real numbers. Given a state $(x_1, x_2, \ldots, x_k)$, a move consists of choosing any i between 1 and $k-1$, inclusive, such that $x_i > x_{i+1}$ and swapping the positions of x_i and x_{i+1}. In other words we move from the state $(x_1, \ldots, x_i, x_{i+1}, \ldots, x_k)$ to the state $(x_1, \ldots, x_{i+1}, x_i, \ldots, x_k)$. The claim is that given a starting state $(x_1, x_2, \ldots, x_k)$, no matter which sequence of swaps are chosen, the result will be the terminal state with the numbers in $(x_1, x_2, \ldots, x_k)$ sorted in increasing order. Furthermore, the number of swaps is also determined by the starting state.

(a) (1 Point) Starting with the state $(5, 6, 2, \pi, \sqrt{2})$, perform Bubble Sort two different ways and confirm that the number of swaps is the same for both.

(b) (2 Points) Prove that the Bubble Sort one-player game is terminating.

(c) (3 Points) Prove that the Bubble Sort one-player game satisfies the diamond condition.

(d) (1 Point) Applying the Diamond Lemma, we now know that the Bubble Sort one-player game is confluent. In other words, for all starting states $(x_1, x_2, \ldots, x_k)$, there is a unique corresponding terminal state. Explain why this corresponding terminal state must be the numbers in $(x_1, x_2, \ldots, x_k)$ sorted in increasing order.

(e) (2 Points) Prove that given a starting state $(x_1, x_2, \ldots, x_k)$, the number of moves to reach the terminal state is always the same and explicitly state what this number is in terms of the starting state. This will conclude the proof of the validity of Bubble Sort.

Solution:

(a) There are many different ways to perform Bubble sort. One of them is:

$$(5, 6, 2, \pi, \sqrt{2}) \to (5, 6, 2, \sqrt{2}, \pi) \to (5, 6, \sqrt{2}, 2, \pi) \to (5, \sqrt{2}, 6, 2, \pi) \to (\sqrt{2}, 5, 6, 2, \pi)$$

$$\to (\sqrt{2}, 5, 2, 6, \pi) \to (\sqrt{2}, 2, 5, 6, \pi) \to (\sqrt{2}, 2, 5, \pi, 6) \to (\sqrt{2}, 2, \pi, 5, 6)$$

Another one is:

$$(5,6,2,\pi,\sqrt{2}) \to (5,2,6,\pi,\sqrt{2}) \to (5,2,\pi,6,\sqrt{2}) \to (5,2,\pi,\sqrt{2},6) \to (2,5,\pi,\sqrt{2},6)$$

$$\to (2,\pi,5,\sqrt{2},6) \to (2,\pi,\sqrt{2},5,6) \to (2,\sqrt{2},\pi,5,6) \to (\sqrt{2},2,\pi,5,6)$$

All sequences of swaps with the given starting state take exactly 8 swaps/moves to reach the terminal state. Alternatively, stating 9 states rather than 8 swaps was also accepted.

(b) Note that for all starting states, the number of pairs of indices i, j with $i < j$ and $x_i > x_j$ (these are called *inversions*) is finite since the number of total pairs i, j with $i < j$ is finite. Every swap removes exactly one inversion and when there are no more inversions, no more swaps are possible, so a sequence of swaps must terminate.

(c) There are two cases. First, let $(x_1, \ldots, x_i, x_{i+1}, \ldots, x_j, x_{j+1}, \ldots, x_k)$ be an arbitrary state with $x_i > x_{i+1}$ and $x_j > x_{j+1}$. We must prove that there is a sequence of moves starting at $(x_1, \ldots, x_{i+1}, x_i \ldots, x_j, x_{j+1}, \ldots, x_k)$ and a sequence of moves starting at $(x_1, \ldots, x_i, x_{i+1}, \ldots, x_{j+1}, x_j, \ldots, x_k)$ that end at the same state. We do so as follows:
$(x_1, \ldots, x_{i+1}, x_i \ldots, x_j, x_{j+1}, \ldots, x_k) \to (x_1, \ldots, x_{i+1}, x_i \ldots, x_{j+1}, x_j, \ldots, x_k)$
and
$(x_1, \ldots, x_i, x_{i+1}, \ldots, x_{j+1}, x_j, \ldots, x_k) \to (x_1, \ldots, x_{i+1}, x_i \ldots, x_{j+1}, x_j, \ldots, x_k)$.

Now, let $(x_1, \ldots, x_i, x_{i+1}, x_{i+2}, \ldots, x_k)$ be an arbitrary state with $x_i > x_{i+1} > x_{i+2}$. We must prove that there is a sequence of moves starting at $(x_1, \ldots, x_{i+1}, x_i, x_{i+2}, \ldots, x_k)$ and a sequence of moves starting at $(x_1, \ldots, x_i, x_{i+2}, x_{i+1}, \ldots, x_k)$ that end at the same state. We do so as follows:
$(x_1, \ldots, x_{i+1}, x_i, x_{i+2}, \ldots, x_k) \to (x_1, \ldots, x_{i+1}, x_{i+2}, x_i, \ldots, x_k) \to (x_1, \ldots, x_{i+2}, x_{i+1}, x_i, \ldots, x_k)$
and
$(x_1, \ldots, x_i, x_{i+2}, x_{i+1}, \ldots, x_k) \to (x_1, \ldots, x_{i+2}, x_i, x_{i+1}, \ldots, x_k) \to (x_1, \ldots, x_{i+2}, x_{i+1}, x_i, \ldots, x_k)$.

An alternative solution is to prove confluence directly and apply Problem (2a) to get the diamond condition.

(d) Suppose for the sake of contradiction that there is a terminal state $(x_1, x_2, \ldots, x_k)$ that is not sorted in increasing order. Then, there exists i such that $x_i \geq x_{i+1}$. However, since the numbers are by definition unique, we have $x_i > x_{i+1}$. Then, these two numbers can be swapped, meaning the state is not terminal, so we arrive at a contradiction.

(e) We will build on the idea from part (b). Since every swap removes exactly one inversion, and the game terminates exactly when the number of inversions reaches zero, the number of swaps in a sequence of moves beginning at the same starting state and ending at a terminal state is exactly equal to the number of inversions in the starting state. Therefore, the total number of swaps will always be the same.

Problem 4. We will now prove the validity of the *Euclidean Algorithm*, an algorithm used to find the greatest common divisor of a set of positive integers. The Euclidean one-player game is defined such that a state in the game is any finite set of positive integers. Given a state $\{n_1, \ldots, n_k\}$, a move in this game consists of choosing a pair n_i and n_j with $n_i < n_j$ and subtracting n_i from n_j. In other words, we move from the state $\{n_1, \ldots, n_k\} = \{n_1, \ldots, n_i, \ldots, n_j, \ldots, n_k\}$ to the state $\{n_1, \ldots, n_i, \ldots, n_j - n_i, \ldots, n_k\}$. For example, starting at the state $\{2, 4, 7\}$, we may move to any one of the following three states: $\{2, 4-2, 7\} = \{2, 2, 7\} = \{2, 7\}$, $\{2, 4, 7-2\} = \{2, 4, 5\}$, and $\{2, 4, 7-4\} = \{2, 4, 3\} = \{2, 3, 4\}$. We claim that given a starting set of positive integers $\{n_1, n_2, \ldots, n_k\}$, no matter which sequence of moves are chosen, we will end up with the singleton set $\{g\}$, where $g = \gcd(n_1, n_2, \ldots, n_k)$.

(a) (1 Point) Perform the Euclidean Algorithm twice starting with the set $\{20, 30, 40, 55\}$ using two different sequences of moves of your choice and verify that, in both cases, the result is the singleton set $\{\gcd(20, 30, 40, 55)\}$.

(b) (2 Points) Prove that the Euclidean one-player game is terminating.

(c) (4 Points) Prove that the Euclidean one-player game satisfies the diamond condition.

(d) (2 Points) Applying the Diamond Lemma, we now know that the Euclidean one-player game is confluent. In other words, for all starting states $\{n_1, n_2, \ldots, n_k\}$, there is a unique corresponding terminal state. Prove that this corresponding terminal state must equal $\{\gcd(n_1, n_2, \ldots, n_k)\}$. This will conclude the proof of the validity of the Euclidean Algorithm.

Solution:

(a) There are many ways to perform the Euclidean Algorithm. One of them is:

$$\{20, 30, 40, 55\} \to \{20, 30, 40, 15\} \to \{20, 30, 10, 15\} \to \{20, 10, 10, 15\} = \{20, 10, 15\} \to \{5, 10, 15\}$$

$$\to \{5, 10, 5\} = \{5, 10\} \to \{5, 5\} = \{5\} = \{\gcd(20, 30, 40, 55)\}$$

Another one is:

$$\{20, 30, 40, 55\} \to \{20, 30, 40, 15\} \to \{5, 30, 40, 15\} \to \{5, 30, 10, 15\} \to \{5, 15, 10, 15\} = \{5, 10, 15\}$$

$$\to \{5, 10, 5\} = \{5, 10\} \to \{5, 5\} = \{5\} = \{\gcd(20, 30, 40, 55)\}$$

(b) Note that the sum of the elements in any state is always positive, finite, and can only decrease after each move since we only ever subtract positive integers. Therefore, the game must terminate. Taking the first example from the solution for part (a), the sum progresses as follows: $145 \to 105 \to 75 \to 45 \to 30 \to 20 \to 15 \to 5$.

(c) To show that the diamond condition holds, we must handle several cases. Without loss of generality we will assume each state only has the elements relevant for the moves from the initial state we will be making.

- Let state $s := \{a, b, c, d\}$ where $a < b$ and $c < d$. Consider the following two states, to which we can move from s: $s' := \{a, b-a, c, d\}$ and $s'' := \{a, b, c, d-c\}$. From both s' and s'', we can get to $s''' := \{a, b-a, c, d-c\}$.
- Let state $s := \{a, b, c\}$ where $a < b < c$. Consider the following two states, to which we can move to from s: $s' := \{a, b-a, c\}$ and $s'' := \{a, b, c-a\}$. From both s' and s'', we can get to $s''' := \{a, b-a, c-a\}$.
- Let state $s := \{a, b, c\}$ where $a < b < c$. Consider the following two states, to which we can move to from s: $s' := \{a, b-a, c\}$ and $s'' := \{a, b, c-b\}$. We then have $s' \to \{a, b-a, c-b+a\} \to \{a, b-a, c-b\} =: s'''$ and $s'' \to s'''$.
- Let state $s := \{a, b, c\}$ where $a < b < c$. Consider the following two states, to which we can move to from s: $s' := \{a, b, c-a\}$ and $s'' := \{a, b, c-b\}$. We then have $s' \to \{a, b-a, c-a\} \to \{a, b-a, c-b\} =: s'''$ and $s'' \to s'''$.

These cases are exhaustive and thus prove that the Euclidean one-player game satisfies the diamond condition.

(d) We first prove that the terminal state has exactly one element. The terminal state cannot have more than one element because if it did, then we could subtract one element from the other, contradicting the fact that the state is terminal. The terminal state cannot be empty because each move in the Euclidean one-player game reduces the size of the set by at most one, and no more moves can be made after the set reaches one element. Therefore, the terminal state must have exactly one element. Next, we note that moves in the Euclidean one-player game do not change

the gcd of the set. This is because if n_i and n_j are divisible by g with $n_i < n_j$, then we may express $n_i = m_i g$ and $n_j = m_j g$, so we have $n_j - n_i = m_j g - m_i g = g(m_j - m_i)$, which is divisible by g. Clearly, the gcd also may not increase. Therefore, the single element in the terminal state is the gcd of the elements of the original set.

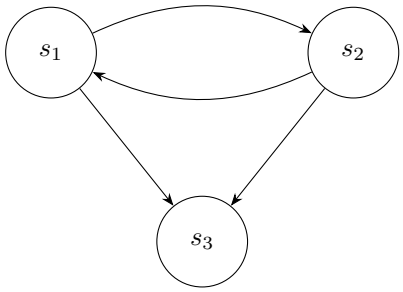
Problem 5. So far, we have only defined confluence for terminating games. Now, we will define it in general. We say a one-player game is *confluent* if choosing a starting state either forces the player to play an infinite game (they can never reach a terminal state no matter what sequence of moves they choose) *or* determines a unique terminal state that the player must end up reaching in a finite number of moves no matter what sequence of moves they choose. We will also define a stronger version of the diamond condition. We say a one-player game satisfies the *extended diamond condition* if whenever there is a state s such that there exists a choice of two states s' and s'' that the player can move to from s, then there also exists a fourth state s''' such that there exists a sequence of moves from s' to s''' as well as a sequence of moves from s'' to s''' such that the two sequences are the *same* length.

(a) (2 Points) Give an example of a non-terminating one-player game that satisfies the regular diamond condition but is not confluent. Explain why your example does not satisfy the extended diamond condition.

(b) (8 Points) Prove that a one-player game that satisfies the extended diamond condition is confluent.

Solution:

(a) Consider the following game:



This satisfies the diamond condition but not the extended diamond condition because the branch $s_1 \to s_2$, s_3 converges to s_3 but the two sequences are not the same length (and similarly for the branch $s_2 \to s_1$, s_3).

(b) Suppose for the sake of contradiction that S is a one-player game that satisfies the extended diamond condition but is not confluent. There are two mutually disjoint cases:

(1) There exists a state that is able to reach two or more different terminal states in a finite number of moves.

(2) There exists a state that is able to reach exactly one terminal state in a finite number of moves but also has an infinite sequence of moves starting at that state.

A contradiction arises from case (1) by the same logic as problem (2b) (but do consider why it is okay that S is not necessarily terminating, unlike in problem (2b)). Then, we only need to deal with case (2).

Supposed case (2) is true. Let s be a state that is able to reach exactly one terminal state in a finite number of moves but also has an infinite sequence of moves starting at that state. Let s_0

be the unique terminal state. We will label all of the states that are able to reach s_0 in a finite number of moves as follows: for any state s' that is able to reach s_0 in a finite number of moves, if n is the shortest sequence of moves from s' to s_0, then we will label s' as order n. We will say that all other states in S have order ∞. Note that s_0 is the unique state of order 0. Also, note that s is able to reach s_0 in a finite number of moves, so it has a label. Then, we will N be the order of s.

Define a complete sequence of moves as a sequence of moves that ends at a terminal state or is of infinite length (that is, a sequence of moves where we keep making moves until we are unable to). We now make the following claim:

> Claim: For all non-negative integers n, if a state s' is of order n, then all complete sequences of moves starting at s' must end at s_0 in exactly n moves.

Note that if we can prove this claim, then a contradiction immediately follows since s is of order N, but there is also an infinite sequence of moves beginning at s.

We will prove the claim using induction.

Base case: n = 0.

If s' is a state of order 0, then s' must be equal to the terminal state s_0. Therefore, the only complete sequence of moves beginning at s' is the empty sequence, which has length 0 and ends at s_0, so the claim is true.

Inductive step:

Suppose the claim is true for $n = k$. Let s' be any state of order $k+1$. Then, there exists a complete sequence of moves of length $k+1$ beginning at s' and ending at s_0. Call this sequence P_1. Let s_1 be the first state after s' along P_1. Then, s_1 must be of order k. Now, consider any complete sequence of moves P_2 beginning at s'. We must show that P_2 ends at s_0 in exactly $k+1$ moves. Let s_2 be the first state after s' along P_2. Note that s_2 is of order at least k. By the extended diamond condition, there exists two paths P_1' and P_2' beginning at s_1 and s_2, respectively, such that P_1' and P_2' both end at some state s_3 and are of the same length l. Let P_3 be any complete sequence of moves beginning at s_3. Let l_3 be the length of P_3. Consider the sequence $P_1' + P_3$, where $+$ denotes concatenation. Note that $P_1' + P_3$ is a complete sequence. By the inductive hypothesis, since s_1 is of order k, we have that $P_1' + P_3$ must terminate at s_0 in exactly k moves. Therefore, $l + l_3 = k$. Then, $P_2' + P_3$ must also terminate at s_0 in exactly k moves, so s_2 is of order k. Applying the inductive hypothesis again, since P_2 is a complete sequence of moves, and the second state of P_2 is s_2, which we have just shown has order k, we have that P_2 must terminate at s_0 in exactly $k+1$ moves.