Hamburger Beiträge
zur Angewandten Mathematik

Logarithmic growth of temporal and spatial complexity in single and double reversals

Julia Sternberg and Andreas Griewank

Nr. 2007-06
April 2007

Logarithmic growth of temporal and spatial complexity in single and double reversals

Julia Sternberg
Department Mathematik,
University of Hamburg
D-20146 Hamburg, Germany,
email: sternberg@math.uni-hamburg.de

Andreas Griewank
Institut für Mathematik,
Humboldt-Universität zu Berlin
D-10099 Berlin, Germany,
griewank@mathematik.hu-berlin.de

Abstract

In the present paper memory efficient implementation of evolutions consisting of two and three alternative sweeps are considered. Two-sweeps evolutions result for instance by a reversal of a program execution needed in automatic differentiation. Three-sweeps evolutions arise through the solution of optimal control problems by Newton's method or through the spatial discretization of PDEs with adaptive FE grids.

The memory efficient implementation of various types of two sweeps evolutions with checkpointing techniques was already discussed in the literature. Nevertheless, in the present paper we introduce and prove new representation for the minimal temporal complexity of single reversals. This new representation gives so far unknown properties of optimal checkpointing strategies and thus provides new insights. After that, we extend the so far known results concerning the logarithmic complexity of single reversal to the double reversal case. In this context we prove in the present paper that the temporal complexity needed to implement a triple-sweep evolution grows as a second power of natural logarithm of a number of time steps representing a sweep length.

Key words. Checkpointing, Nested Checkpointing, Dynamic Programming, Logarithmic Growth of Complexity

1 Single Reversal

The key difficulty in adjoining large evaluation programs is their reversal within a reasonable amount of memory. If the total number of intermediate states is to be stored, the memory requirement will be proportional to run-time needed to run a program and can be enormous. A remedy for these memory problems is the so-called **checkpointing technique**. There are many theoretical investigations regarding this technique. Definitions, notations and basic results are introduced in this section.

1.1 Notations and Definitions

Many mathematical applications contain a **forward simulation**. An example for a forward simulation is a time discretization of an ODE or a PDE. Let us denote a forward simulation as F.

A certain sequence of intermediate states is to be computed during the evaluation of a forward simulation F. Each action from a previous intermediate state to the next intermediate state can be seen as a **time step** F_i. Therefore, without loss of generality, we can say that the evaluation of a given forward simulation F can be represented as a computation of l time steps $F_i,\ 1 \leq i \leq l$. A certain number of assignments are to be combined to a single step. Figure 1 illustrates this situation. Time steps $F_i,\ 1 \leq i \leq l$, are depicted by arrows.

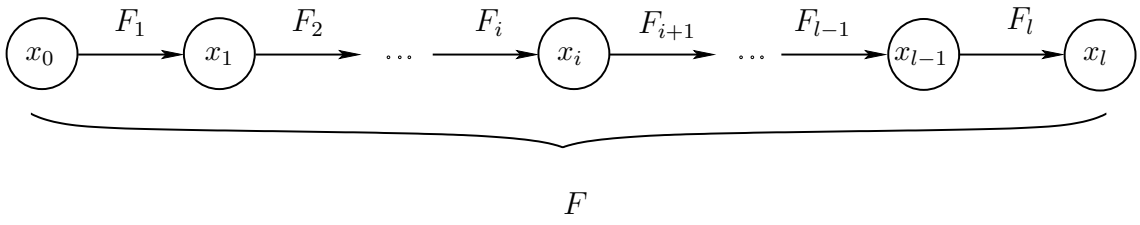

Figure 1: Forward simulation F

An intermediate state x_i is a large dimensional vector. The intermediate state x_i can be evaluated if the time step F_i is applied to the previous state x_{i-1}. The time step F_i evaluates the intermediate state x_i using only data from the previous state x_{i-1}.

If we solve an adjoint problem associated with a given forward simulation F, or if we apply the adjoint mode of the algorithmic or automatic differentiation, it is necessary to reverse this forward simulation F.

The basic method to invert a forward simulation F is to store all intermediate states $x_i,\ 0 \leq i \leq l$, on a sequential data file and to restore them if required. In the literature this method is called the **basic approach**.

The memory requirement for the basic approach is proportional to the run-time of a forward simulation F, i.e. to the number l of time steps. Therefore, it could be huge. Practical applications of the basic approach are restricted in spite of the fact that the memory capacity grows. This is the sort of problems, checkpointing deals with.

Another approach for the reversal of a forward simulation F does not require to store all intermediate states, but a smaller number of them. Stored intermediate states are called **checkpoints**. Consequently, the method is called **checkpointing**. Checkpointing is introduced and investigated in [1, 3, 4, 10]. Depending on the available memory capacity, we can store only a small number of intermediate states at the same time. Therefore, this approach enormously reduces the memory requirement.

Using checkpointing, data of selected intermediate states is to be stored as checkpoints. Thus, during the reversal of a forward simulation F, the evaluation of some time steps as well as some intermediate states could occur several times. Consequently, the evaluation procedure has to be carried out repeatedly. There is no need to restart a forward simulation F from the initial intermediate state. It can be restarted from a suitable checkpoint.

In order to provide the reversal or the adjoint procedure of a forward simulation F, we expand it by adding to each time step F_i an additional **adjoint** step $\bar{F}_i$. The resulting adjoint simulation is denoted $\bar{F}$. In the following, we denote a forward simulation F an primal sweep and an adjoint simulation $\bar{F}$ an adjoint sweep. Therefore, corresponding time steps and intermediate states are denoted primal steps or primal states and adjoint steps or adjoint states, for primal and adjoint sweeps, respectively. The situation is depicted in

Figure 2, which is to be seen as an extension of Figure 1. Here, to each primal step F_i an

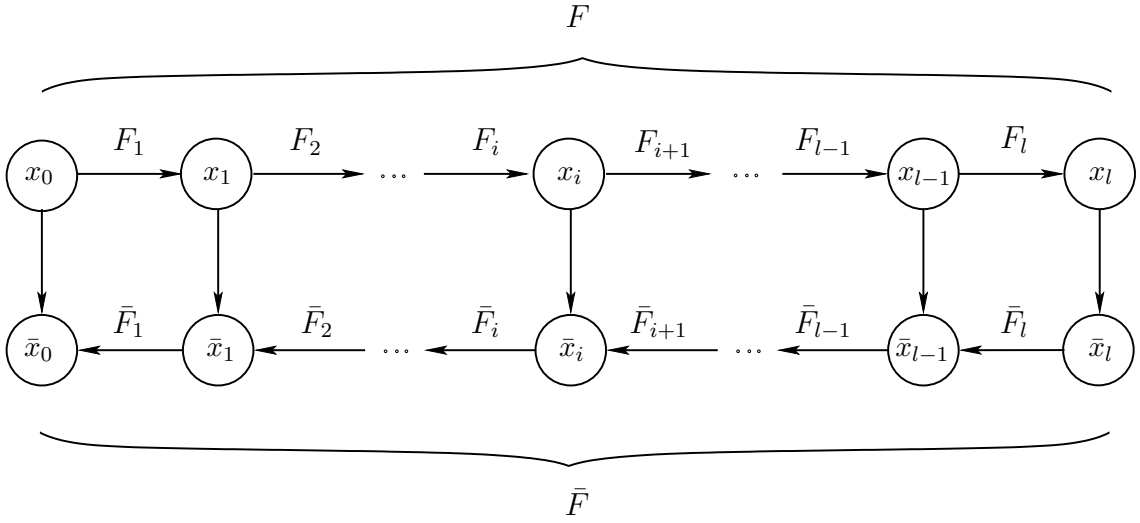

Figure 2: Forward simulation F and its associated adjoint simulation $\bar{F}$

adjoint step $\bar{F}_i$ is associated. A corresponding adjoint state, associated with the primal state x_i, is represented by the counter $\bar{x}_i$. The adjoint state $\bar{x}_{i-1}$ can be evaluated after the primal step F_i is applied to the primal state x_{i-1} in order to evaluate the consecutive primal state x_i. Data, produced during this evaluation, is to be stored on a tape. After that, the adjoint state $\bar{x}_{i-1}$ can be evaluated using data of the consecutive adjoint state $\bar{x}_i$ and recently stored data from the evaluation F_i. Without loss of generality, we combine these assignments to the adjoint step $\bar{F}_i$. Thus, the adjoint step $\bar{F}_i$ evaluates the adjoint state $\bar{x}_{i-1}$ using data from the consecutive adjoint state $\bar{x}_i$ as well as data produced during the evaluation of the primal state x_i. The adjoint step $\bar{F}_i$ can be evaluated only if data of the primal state x_{i-1} and data of the consecutive adjoint state $\bar{x}_i$ is available. To simplify notations we associate a pair of a forward and adjoint simulations or sweeps, containing l steps, with a double sweep evolution denoted by $\mathcal{E}_{2\times l}$.

We introduce evaluation costs, i.e. the computational effort for intermediate steps of different sweeps as

$$t = TIME(F_i), \quad \bar{t} = TIME(\bar{F}_i), \quad 1 \leq i \leq l. \tag{1}$$

In this paper we assume uniform evaluation costs within a corresponding sweep. For investigations concerning single reversal with non-uniform costs see [7, 11, 12].

The reversal of a forward simulation F can be interpreted as a sequence of assignments. They are to combine into appropriate actions. Each action defines which primal step is to evaluate, which primal state is to store or restore from a corresponding checkpoint. A certain combination of actions represents a **reversal schedule**. More formally, use the following definition.

Definition 1.1. (**Reversal Schedule S**). Consider a double sweep evolution $\mathcal{E}_{2\times l}$, so that each sweep of it comprises l primal steps F_i, $1 \leq i \leq l$. Let $s \in \mathbb{N}$ checkpoints be available each of which can accommodate data of a single primal state. l denotes the number of the currently final primal step, i denotes the current state and j denotes the number of stored checkpoints. Initially, $i = i_0 \geq 0$, $j = j_0 \geq 0$, and $i < l \leq \infty$. Then, a reversal schedule **S** consists of a finite sequence of the following actions:

A = If $i < l-1$, carry out the primal step F_{i+1} and set $i = i+1$.
W_k = Write data of the current primal state into the checkpoint $j_0 + k$.
R_k = Read out an primal state from the checkpoint $j_0 + k$ and set the counter i to its number.
U = If $0 \leq l - 1 = i$, carry out the adjoint step $\bar{F}_{i+1}$ and set $l = l - 1$, else terminate.

We normally assume that all reversal schedules terminate successfully in that $l = 0$ at the end. We then say that the schedule is admissible for the given combination (l, s). Without loss of generality, we assume that each admissible reversal schedule $\mathbf{S}$ begins with the action R_0, so that the initial primal state is read out from the checkpoint j_0. If an action A is carried out up to k times, it is denoted A^k.

Note that some results from [4, 5, 11] are already incorporated in Definition 1.1. In particular, a reversal schedule $\mathbf{S}$ is defined in such a manner that each adjoint step $\bar{F}_i,\ 1 \leq i \leq l$, is evaluated exactly once. Therefore, the overall run-time requirement for the evaluation of all adjoint steps $\bar{F}_i,\ 1 \leq i \leq l$, is defined by $l\bar{t}$. Furthermore, for each reversal schedule $\mathbf{S}$, $l_{max}(\mathbf{S})$ denotes the maximum of the achieved index $(i - i_0)$ and $s_{max}(\mathbf{S})$ denotes the maximum of stored checkpoints, i.e. the maximum of $(j - j_0 + 1)$. For an admissible reversal schedule $\mathbf{S}$ the **repetition numbers** $r_i \equiv r(i)$ defined by the function

$$r : [1, l] \to \mathbb{N}, \tag{2}$$

count how often the ith primal step is evaluated during the execution of the reversal schedule $\mathbf{S}$.

Figure 3 illustrates an example of a reversal schedule $\mathbf{S}$ with $l_{max}(\mathbf{S}) = 10$ and $s_{max}(\mathbf{S}) = 3$ for $i_0 = 0$ and $j_0 = 0$. The temporal complexity for primal and adjoint steps is given by $t = 1$ and $\bar{t} = 2$, respectively. Primal steps $F_i,\ 1 \leq i \leq 10$, are plotted along the vertical y-axis,

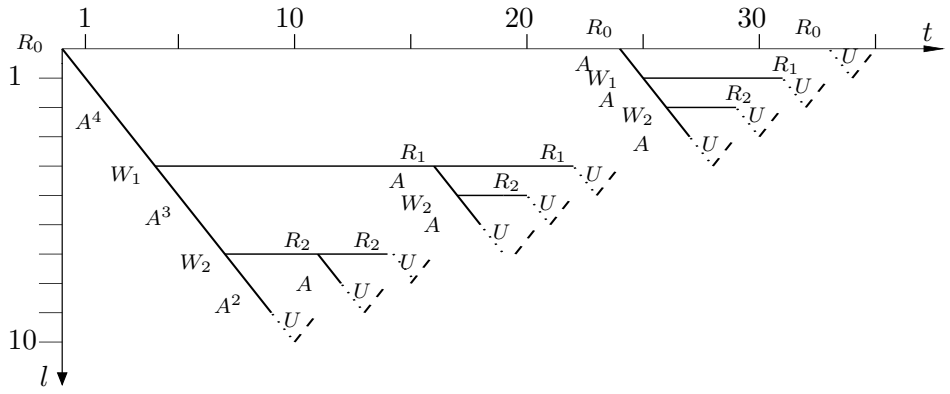

Figure 3: Reversal schedule $\mathbf{S}$ with $l_{max}(\mathbf{S}) = 10$ and $s_{max}(\mathbf{S}) = 3$

whereas time required for the evaluation of single actions is plotted along the horizontal x-axis. Time is measured in t-units. Horizontal lines in Figure 3 represent checkpoints. Since the initial checkpoint contains data of the primal state x_0 and lies on the computational axis, this axis represents a checkpoint itself.

The execution of the reversal schedule $\mathbf{S}$ in Figure 3 can be described as follows. Firstly, the action R_0 is performed, i.e. the initial primal state x_0 is read out from the first checkpoint

$j_0 = 0$. After that, the four primal steps F_1, F_2, F_3, and F_4 are evaluated consecutively, which can be described by the action A^4. Then, the primal state x_4 is stored into the second checkpoint by the action W_1. The following 3 primal steps F_5, F_6, and F_7 are carried out by the action A^3. The primal state x_7 is stored into the third checkpoint by the action W_2. The following two primal steps F_8 and F_9 are carried out by the action A^2 consecutively. After that, the action U is applied for the first time, so that the adjoint step $\bar{F}_{10}$ is evaluated. Then, data of the primal state x_7 is read out from the third checkpoint by the action R_2. Using this data, the primal step F_8 is evaluated by the action A. Now, it is possible to perform the adjoint step $\bar{F}_9$ using the action U. After that, the same principle is applied several times until all adjoint steps $\bar{F}_1, ..., \bar{F}_{10}$ are performed successively.

The reversal schedule $\mathbf{S}$ in Figure 3 can be represented by the following sequence of actions:

$$
\begin{aligned}
\mathbf{S} \;=\; & R_0 \triangleright A^4 \triangleright W_1 \triangleright A^3 \triangleright W_2 \triangleright A^2 \triangleright U \triangleright R_2 \triangleright A \triangleright R_2 \triangleright U \triangleright \\
& \triangleright R_1 \triangleright A \triangleright W_2 \triangleright A \triangleright U \triangleright R_2 \triangleright U \triangleright R_1 \triangleright U \triangleright R_0 \triangleright A \triangleright \\
& \triangleright W_1 \triangleright A \triangleright W_2 \triangleright U \triangleright R_2 \triangleright U \triangleright R_1 \triangleright U \triangleright R_0 \triangleright U.
\end{aligned} \tag{3}
$$

The operator '$\triangleright$' between two actions can be interpreted as a successive performance of these actions from left to the right.

The reversal schedule $\mathbf{S}$ in Figure 3 can be executed using 35 t-units. During the evaluation of $\mathbf{S}$, memory for three checkpoints is required. Moreover, memory for the storage of data produced during a single primal step $F_i,\ 1 \leq i \leq 10$, which is required for the performing of a corresponding adjoint step $\bar{F}_i,\ 1 \leq i \leq 10$, is needed. If the basic approach is applied, merely 20 t-units will be needed to perform ten adjoint steps $\bar{F}_i,\ 1 \leq i \leq 10$, but all primal states $x_i,\ 0 \leq i \leq 10$, as well as data produced during all single primal steps $F_i,\ 1 \leq i \leq 10$, which is required for the performance of corresponding adjoint steps $\bar{F}_i,\ 1 \leq i \leq 10$, have to be stored.

For the reversal schedule $\mathbf{S}$ in Figure 3 the corresponding repetition profile is given by

$$
\mathbf{r}(\mathbf{S}) = \langle 2, 2, 2, 1, 2, 2, 1, 2, 1, 0 \rangle^{\top}. \tag{4}
$$

This profile is illustrated in Figure 4. The maximal repetition number $r_{max}(\mathbf{S})$ reaches two.

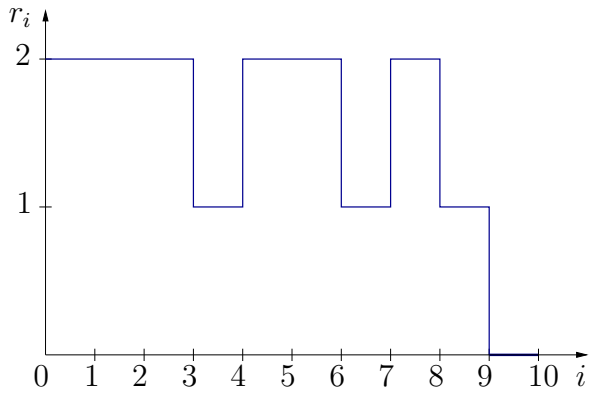

Figure 4: Repetition profile for the reversal schedule $\mathbf{S}$ in Figure 3

The temporal complexity of a reversal schedule $\mathbf{S}$, i.e. the run-time effort, which is required

to execute the reversal schedule $\mathbf{S}$ applied to a double sweep evolution $\mathcal{E}_{2\times l}$, is defined by

$$\mathbf{T}(\mathbf{S}) = \sum_{i=1}^{l} r_i, \qquad \text{if} \qquad l \leq l_{max}(\mathbf{S}). \tag{5}$$

In the definition of $\mathbf{T}(\mathbf{S})$ we have omitted the temporal complexity, needed to perform adjoint states, since this complexity is always constant, namely $l\bar{t}$. Thus, to characterize the temporal complexity of a reversal schedule $\mathbf{S}$, it is sufficient only to count the number of primal steps evaluated during its execution. For the example of a reversal schedule $\mathbf{S}$ in Figure 3 the temporal complexity $\mathbf{T}(\mathbf{S})$ attains $\mathbf{T}(\mathbf{S}) = 15$.

Now, a naturally raised question is, which reversal schedule from the total amount of possible reversal schedules for a forward simulation F and s available checkpoints causes a minimal number of primal steps through its execution. The second question is, how to construct an appropriate reversal schedule. In order to find answers for these questions, we introduce some needful characteristic values first.

1.2 Decomposition of Reversal Schedules

To establish explicit formulae for quantities introduced in the previous section reversal schedules will be decomposed into smaller substructures which will be considered separately. To make this, the following assertion proved in [4, 11] is necessary.

Lemma 1.2. *(***Checkpoint Persistence).** *Consider a reversal schedule* $\mathbf{S}$ *with* $l_{max}(\mathbf{S}) = l$ *and* $s_{max}(\mathbf{S}) = s$. $\mathbf{T}(\mathbf{S})$ *denotes its evaluation cost resulting through the implementation of the double sweep evolution* $\mathcal{E}_{2\times l}$. *Then,* $\mathbf{S}$ *can be modified without increasing the cost* $\mathbf{T}(\mathbf{S})$*, so that the following properties hold. After the checkpoint writing* W_j*, storing the primal state* x_i *into the checkpoint* j*, the next action* W_j *occurs only when* l *has already been reduced to* i *or below. Moreover, until that time no actions involving the primal states between* x_0 *and* x_i *are taken.*

Using the checkpoint persistence, each reversal schedule $\mathbf{S}$ can be decomposed into two subschedules $\mathbf{S}^1$ and $\mathbf{S}^2$. This can be done by storing the primal state $x_{\check{l}}$ into the checkpoint $(j_0 + 1)$ using the action W_1. The reversal schedule $\mathbf{S}^1$ represents a set of actions regarding the primal steps $F_1, ..., F_{\check{l}}$, whereas the reversal schedule $\mathbf{S}^2$ contains actions regarding the remaining primal steps $F_{\check{l}+1}, ..., F_l$. Obviously, $l_{max}(\mathbf{S}^1) = \check{l}$ and $l_{max}(\mathbf{S}^2) = l - \check{l}$. This decomposition is illustrated in Figure 5 and can be defined by the following binary composition

$$\mathbf{S} = \mathbf{S}^1 \circ \mathbf{S}^2. \tag{6}$$

The operator $\circ$ in (6) is to be interpreted in the following way: actions representing the reversal schedule $\mathbf{S}^1$ affect primal steps $F_1, ..., F_{\check{l}}$; actions representing the reversal schedule $\mathbf{S}^2$ affect primal steps $F_{\check{l}+1}, ..., F_l$. This operator $\circ$ is certainly not commutative. Also, in contrast to $\triangleright$, the operator $\circ$ is not associative. This can be seen in the following example.

Example 1.3. Suppose, we have three different reversal schedules $\mathbf{S}^1$, $\mathbf{S}^2$, and $\mathbf{S}^3$. If the operator $\circ$ is associative, then, the following equation will be valid:

$$(\mathbf{S}^1 \circ \mathbf{S}^2) \circ \mathbf{S}^3 = \mathbf{S}^1 \circ (\mathbf{S}^2 \circ \mathbf{S}^3). \tag{7}$$

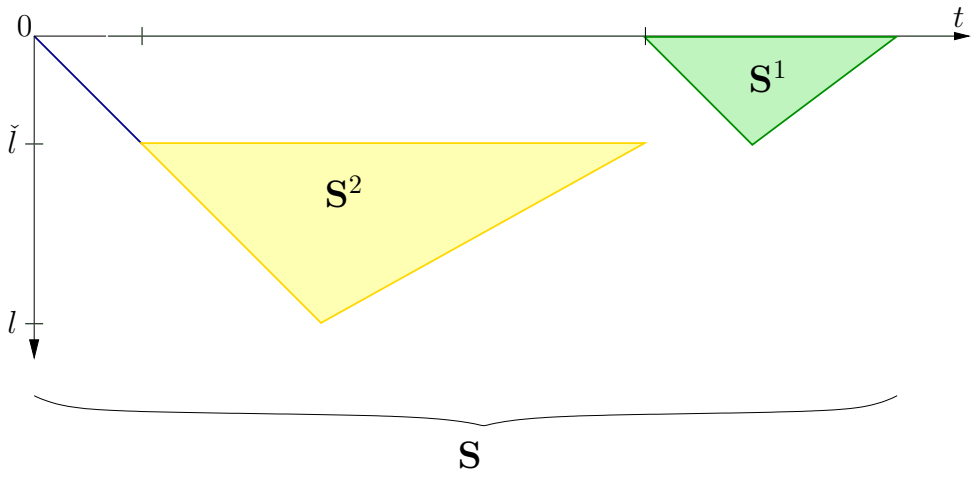

Figure 5: Decomposition of reversal schedule $\mathbf{S} = \mathbf{S}^1 \circ \mathbf{S}^2$

Firstly, we denote $s_{max}(\mathbf{S}^i)$, $i = 1, 2, 3$, the maximal number of checkpoints available throughout the execution of a reversal schedule $\mathbf{S}^i$, $i = 1, 2, 3$. Therefore, using the definition of the binary decomposition (6) above, we obtain the following relations from the left hand side of the equation (7).

$$\begin{array}{rcl} s_{max}(\mathbf{S}^2) & = & s_{max}(\mathbf{S}^1) - 1, \\ s_{max}(\mathbf{S}^3) & = & s_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) - 1 \quad = \quad s_{max}(\mathbf{S}^1) - 1. \end{array} \tag{8}$$

If we apply similar considerations to the right-hand side of the equation (7), we obtain the following relations.

$$\begin{array}{rcl} s_{max}(\mathbf{S}^3) & = & s_{max}(\mathbf{S}^2) - 1, \\ s_{max}(\mathbf{S}^2 \circ \mathbf{S}^3) & = & s_{max}(\mathbf{S}^2) \quad = \quad s_{max}(\mathbf{S}^1) - 1, \\ s_{max}(\mathbf{S}^3) & = & s_{max}(\mathbf{S}^1) - 2. \end{array} \tag{9}$$

Therefore, if we compare the two last equations (8) and (9), then, we obtain that if (7) is true, i.e. if the operator $\circ$ is associative, then, the following equation will be satisfied:

$$s_{max}(\mathbf{S}^1) - 1 = s_{max}(\mathbf{S}^1) - 2. \tag{10}$$

Obviously, the relation (10) does not hold. Therefore, the operator $\circ$ is not associative.

An example for a reversal schedule $\mathbf{S}$ given by the decomposition $\mathbf{S} = (\mathbf{S}^1 \circ \mathbf{S}^2) \circ \mathbf{S}^3$ is shown in Figure 6. Another example for a reversal schedule $\mathbf{S}$ given by the decomposition $\mathbf{S} = \mathbf{S}^1 \circ (\mathbf{S}^2 \circ \mathbf{S}^3)$ is shown in Figure 7. Comparing Figure 6 with Figure 7, we can see that the reversal schedules $\mathbf{S}$ given by these figures are different, although the subschedules $\mathbf{S}^1$, $\mathbf{S}^2$, and $\mathbf{S}^3$, used in these decompositions are assumed to be the same. Therefore, comparing Figure 6 with Figure 7 we can also see that the operator $\circ$ is not associative.

□

The decomposition (6) implies the following relations

$$l_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) = l_{max}(\mathbf{S}^1) + l_{max}(\mathbf{S}^2), \tag{11}$$

$$s_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) = \max\left\{s_{max}(\mathbf{S}^1), s_{max}(\mathbf{S}^2) + 1\right\}, \tag{12}$$

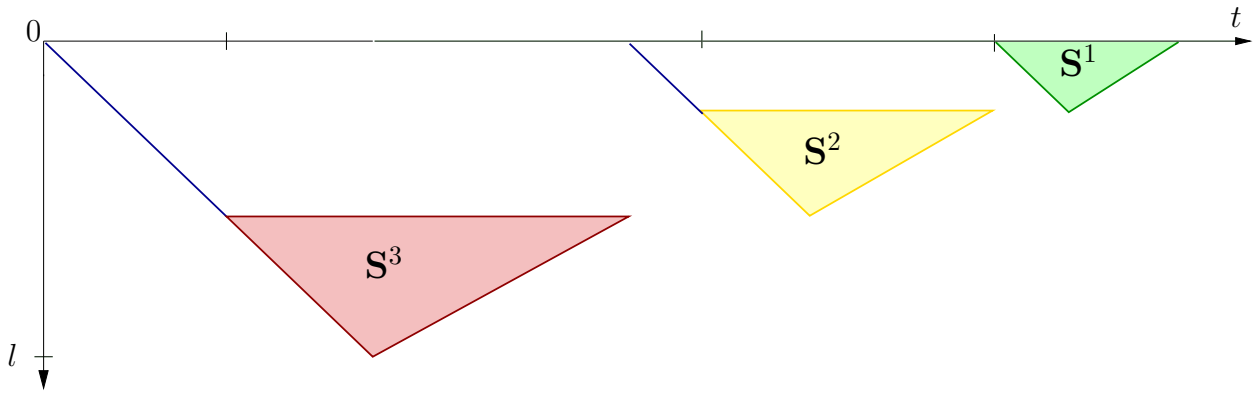

Figure 6: Decomposition of reversal schedule $\mathbf{S} = (\mathbf{S}^1 \circ \mathbf{S}^2) \circ \mathbf{S}^3$

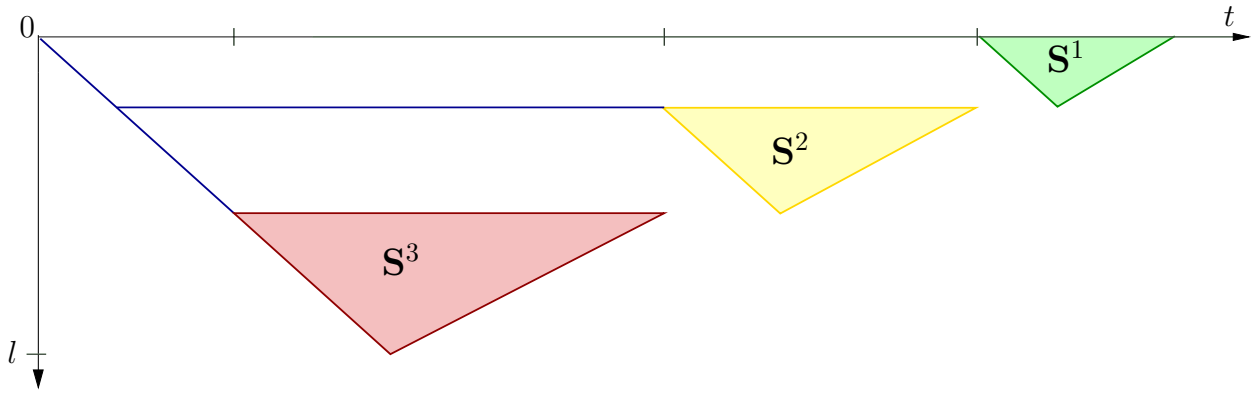

Figure 7: Decomposition of reversal schedule $\mathbf{S} = \mathbf{S}^1 \circ (\mathbf{S}^2 \circ \mathbf{S}^3)$

$$r_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) = \max\left\{r_{max}(\mathbf{S}^1) + 1, r_{max}(\mathbf{S}^2)\right\}. \tag{13}$$

The property (12) concerning the maximal number of checkpoints was already used in Example 1.3. Using the properties (11) - (13), one can evaluate explicit formulae for l_{max}, s_{max}, and r_{max}, which will be the topic of next sections.

1.3 Maximal Length $\mathbf{l_{max}(s, r)}$

We denote $l_{max}(s, r)$ the maximal number of primal steps which can be reversed using s available checkpoints, so that each primal step F_i, $1 \le i \le l$, is evaluated up to r times during this reversal. $l_{max}(s, r)$ can be defined by

$$l_{max}(s, r) = \max\left\{l_{max}(\mathbf{S}) : \ s_{max}(\mathbf{S}) \le s, \ r_{max}(\mathbf{S}) \le r\right\}. \tag{14}$$

For the evaluation of the maximal length $l_{max}(s, r)$ for given values s and r we apply the decomposition of a reversal schedule $\mathbf{S}$ according to (6). Therefore, we obtain

$$\begin{aligned}
l_{max}(s,r) &= \max\left\{l_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) : \ s_{max}(\mathbf{S}^1) \le s, \ s_{max}(\mathbf{S}^2) \le s-1, \right. \\
&\qquad \left. r_{max}(\mathbf{S}^1) \le r-1, \ r_{max}(\mathbf{S}^2) \le r\right\} = \\
&= \max\left\{l_{max}(\mathbf{S}^1) + l_{max}(\mathbf{S}^2) : \ s_{max}(\mathbf{S}^1) \le s, \ s_{max}(\mathbf{S}^2) \le s-1, \right. \\
&\qquad \left. r_{max}(\mathbf{S}^1) \le r-1, \ r_{max}(\mathbf{S}^2) \le r\right\} = \\
&= \max\left\{l_{max}(\mathbf{S}^1) : \ s_{max}(\mathbf{S}^1) \le s, \ r_{max}(\mathbf{S}^1) \le r-1 \ \right\} + \\
&+ \max\left\{l_{max}(\mathbf{S}^2) : \ s_{max}(\mathbf{S}^2) \le s-1, \ r_{max}(\mathbf{S}^2) \le r \ \right\} = \\
&= l_{max}(s, r-1) + l_{max}(s-1, r).
\end{aligned} \tag{15}$$

The relation (15) is illustrated in Figure 8.

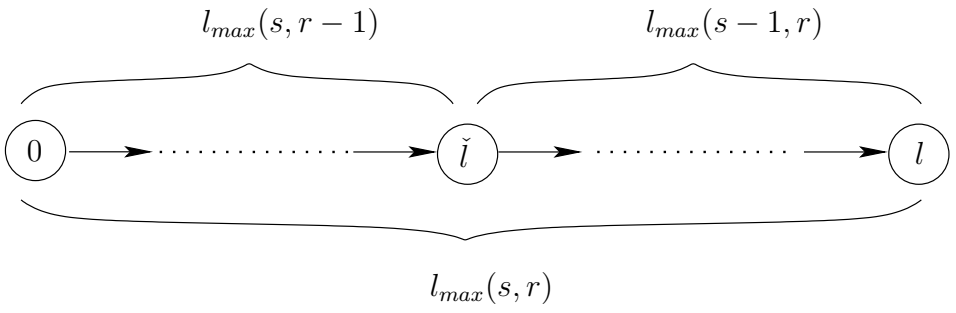

Figure 8: Decomposition of a step sequence according to (15)

Values of $l_{max}(s, r)$ for $s = 1$ and $r = 1$ can be determined using Definition 1.1 of reversal schedules. Thus,

$$l_{max}(1, r) = r + 1, \qquad l_{max}(s, 1) = s + 1, \qquad s, r \ge 1. \tag{16}$$

The equations (15) and (16) yield the following result from [4]. Here and throughout this paper, we use the abbreviation $\beta(s, r) = \frac{(s+r)!}{s!\, r!}$.

Lemma 1.4. *(***Binomial Rule).** *The maximal number of steps $l_{max}(s,r)$ can be evaluated explicitly by*

$$l_{max}(s,r) = \beta(s,r) = \binom{s+r}{s} = \frac{(s+r)!}{s!\,r!} \approx \begin{cases} \exp(r)/\sqrt{r} & \text{if} \quad s \sim r, \\ s^r & \text{if} \quad s \gg r = const, \\ r^s & \text{if} \quad r \gg s = const. \end{cases} \tag{17}$$

1.4 Minimal Temporal Complexity for Single Reversal Schedules

Minimal number of primal steps which represents the minimal temporal complexity, needed to implement a double seep evolution $\mathcal{E}_{2\times l}$ with up to s available checkpoints is defined by

$$\mathbf{T}_{bin}(l,s) = \min_{\mathbf{S}} \big\{ \mathbf{T}(\mathbf{S}),\ l \le l_{max}(\mathbf{S}) \text{ and } s_{max}(\mathbf{S}) \le s \big\}. \tag{18}$$

To derive an explicit representation for $\mathbf{T}_{bin}(l,s)$ the following theorem was proved into different ways in [4, 7, 11].

Theorem 1.5. (Minimal Evaluation Cost). *The function $\mathbf{T}_{bin}(l,s)$, which denotes the minimal number of primal steps needed to reverse l primal steps with up to s checkpoints accommodated at any time, yields the form*

$$\mathbf{T}_{bin}(l,s) = \min_{0<\check{l}<l} \{ \check{l} + \mathbf{T}_{bin}(l-\check{l}, s-1) + \mathbf{T}_{bin}(\check{l}, s) \}. \tag{19}$$

This implies

$$\mathbf{T}_{bin}(l,s) = rl - \beta(s+1, r-1), \tag{20}$$

with $r = r_{bin}(s,l)$, being the unique integer satisfying

$$\beta(s, r-1) \ < \ l \ \le \ \beta(s,r). \tag{21}$$

Theorem 1.5 implies the following corollary, which gives a new representation of the minimal evaluation cost $\mathbf{T}_{bin}(l,s)$ and a new proof for its explicit formula.

Corollary 1.6. Let the assumptions of Theorem 1.5 be satisfied. Then, the function $\mathbf{T}_{bin}(l,s)$ can be evaluated by

$$\mathbf{T}_{bin}(l,s) = \sum_{i=1}^{l} r(s,i), \tag{22}$$

with $r(s,i) = r_{bin}(s,i),\ 1 \le i \le l$, being the unique integers satisfying

$$\beta(s, r(s,i)-1) \ < \ i \ \le \ \beta(s, r(s,i)). \tag{23}$$

Proof. Relation (22) can be proved by induction.

Trivial Case I ($s = 1$). To achieve the minimal number $\mathbf{T}_{bin}(l,1)$ of time step evaluations, we can explicitly derive the corresponding reversal schedule: The only checkpoint available must be set to the initial primal state x_0. For the first Reverse action U in Definition 1.1, the primal steps F_j, $1 \le j \le l-1$ have to be evaluated. To perform the next Reverse action

U, the primal steps F_j, $1 \le j \le l-2$ have to be evaluated and so on. Hence, for the overall reversal we obtain

$$\mathbf{T}_{bin}(l,1) = l-1+(l-2)+\cdots+1+0 = \sum_{i=1}^{l}(i-1) = \sum_{i=1}^{l} r(1,i),$$

since $r(1,i) = i-1$ for $1 \le i \le l$.

Trivial Case II ($1 < l \le s+1$).

For this choice of l and s we obtain

$$\beta(s,0) = 1 < l \le s+1 = \beta(s,1) \qquad \Longleftrightarrow \qquad r = 1.$$

The s checkpoints must be set to all primal states except for the last two states. For the first reverse step a minimum of $l-1$ time steps is needed. To perform other reverse steps, no time steps are necessary. Therefore, the minimal number of time steps is equal to

$$\mathbf{T}_{bin}(l,s) = l-1 = \underbrace{0+1+1+...+1}_{l} = \sum_{i=1}^{l} r(s,i),$$

since $r(s,1) = 0$ and $r(s,i) = 1$ for $2 \le i \le l \le s+1$.

Induction step over l. The numbers s and l are given. Assume that the assertion (22) is true for all $(s,\tilde{l})$ with $\tilde{l} \le l$. Now, it will be shown that the property (22) is valid for the pair $(s, l+1)$, i.e.

$$\mathbf{T}_{bin}(l+1,s) = \sum_{i=1}^{l+1} r(s,i). \tag{24}$$

Obviously, $\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = \sum_{i=1}^{l+1} r(s,i) - \sum_{i=1}^{l} r(s,i) = r(s,l+1)$.

Therefore, in order to prove (24) is sufficient, it is to be shown that $\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = r(s,l+1)$, with $\mathbf{T}_{bin}(l+1,s)$ and $\mathbf{T}_{bin}(l,s)$ evaluated using (20). Thus, applying (20), we obtain

$$\begin{aligned}&\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = r(s,l+1)(l+1) - \\ &-\beta(s+1, r(s,l+1)-1) - r(s,l)l + \beta(s+1, r(s,l)-1).\end{aligned}$$

Here, consider two different cases:

1st case: $r(s,l+1) = r(s,l)$. Therefore,

$$\begin{aligned}&\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = r(s,l+1)(l+1) - \\ &-\beta(s+1, r(s,l+1)-1) - r(s,l)l + \beta(s+1, r(s,l)-1) = \\ &= r(s,l)(l+1) - r(s,l)l = r(s,l) = r(s,l+1),\end{aligned}$$

which proves the assertion of Corollary 1.6 for this case.

2nd case: $r(s,l+1) = r(s,l)+1$. Note that if $r(s,l+1) = r(s,l)+1$, relation (21) implies that $l = \beta(s, r(s,l))$. Therefore,

$$\begin{aligned}
&\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = r(s,l+1)(l+1) - \\
&-\beta(s+1, r(s,l+1)-1) - r(s,l)l + \beta(s+1, r(s,l)-1) = \\
&= (r(s,l)+1)(l+1) - r(s,l)l - \beta(s+1, r(s,l)) + \beta(s+1, r(s,l)-1) = \\
&= (l+1) + r(s,l) - \beta(s, r(s,l)) = (l+1) + r(s,l) - l = \\
&= r(s,l) + 1 = r(s,l+1).
\end{aligned}$$

This proves the assertion of Corollary 1.6 for the second case. Here, we use the relation

$$\beta(s+1, r(s,l)) - \beta(s+1, r(s,l)-1) = \beta(s, r(s,l)).$$

This is straightforwardly proved applying the definition of the function β in (17).

□

Alternatively to the proof of Corollary 1.6 given above, where the equivalence between the equation (20) and the equation (22) is shown, we can straightforwardly prove Corollary 1.6. In this case, there is no need to assume the relation (20) to be satisfied. The equality (22) directly follows from the partition (19) and basic properties of repetition numbers.

Alternative proof: The minimal evaluation cost $\mathbf{T}_{bin}(l,s)$, defined by the formula (19), can be evaluated explicitly using (22). Here, we show the equivalence between the equation (19) and the equation (22). This equivalence can also be proved by induction.

For the *Trivial Case* I ($s = 1$) and the *Trivial Case* II ($1 < l \leq s+1$) we obtain

$$\mathbf{T}_{bin}(l,s) = \sum_{i=1}^{l} r(s,i).$$

In order to show this formula, we can use the same considerations as in the first proof, because for these trivial cases the representation (20) is not assumed. Relation (22) was shown only using basic properties of reversal schedules.

Induction step over l. The numbers s and l are given. Assume that the assertion (22) is true for all $(s, \tilde{l})$ with $\tilde{l} \leq l$. Now, it will be shown that the property (22) is valid for the pair $(s, l+1)$, i.e.

$$\mathbf{T}_{bin}(l+1,s) = \sum_{i=1}^{l+1} r(s,i). \tag{25}$$

To show this relation we have to prove

$$\mathbf{T}_{bin}(l+1,s) - \mathbf{T}_{bin}(l,s) = \sum_{i=1}^{l+1} r(s,i) - \sum_{i=1}^{l} r(s,i) = r(s,l+1), \tag{26}$$

with $\mathbf{T}_{bin}(l+1,s)$ and $\mathbf{T}_{bin}(l,s)$ being evaluated using (19). Assume that an optimal value $\check{l}^\star$ is determined, so that

$$\mathbf{T}_{bin}(l,s) = \check{l}^\star + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star, s). \tag{27}$$

Then, due to Lemma 1.7 to be proved later on, an optimal value $\check{m}^\star$ for $m = l+1$ has two possibilities to choose from, namely $\check{m}^\star = \check{l}^\star$ or $\check{m}^\star = \check{l}^\star + 1$. Then, the optimal evaluation cost $\mathbf{T}_{bin}(l+1, s)$ is evaluated by

$$\mathbf{T}_{bin}(l+1, s) = \check{m}^\star + \mathbf{T}_{bin}(l+1-\check{m}^\star, s-1) + \mathbf{T}_{bin}(\check{m}^\star, s), \tag{28}$$

with an appropriate value $\check{m}^\star$. Therefore, in order to prove (26), we have to find the difference between the equations (28) and (27) for two cases of $\check{m}^\star$.

1st case: $\check{m}^\star = \check{l}^\star$. Therefore,

$$\begin{gathered}\mathbf{T}_{bin}(l+1, s) - \mathbf{T}_{bin}(l, s) = \check{l}^\star + \mathbf{T}_{bin}(l+1-\check{l}^\star, s-1) + \\ + \mathbf{T}_{bin}(\check{l}^\star, s) - \check{l}^\star - \mathbf{T}_{bin}(l-\check{l}^\star, s-1) - \mathbf{T}_{bin}(\check{l}^\star, s) = \\ = \mathbf{T}_{bin}(l+1-\check{l}^\star, s-1) - \mathbf{T}_{bin}(l-\check{l}^\star, s-1) = \\ = r(s-1, l+1-\check{l}^\star) = r(s, l+1).\end{gathered} \tag{29}$$

Since $1 < \check{l}^\star < l-1$ implies $l-\check{l}^\star+1 < l$, we obtain $\mathbf{T}_{bin}(l+1-\check{l}^\star, s-1) - \mathbf{T}_{bin}(l-\check{l}^\star, s-1) = r(s-1, l+1-\check{l}^\star)$ due to the induction assumption. The last equality in (29) is because of Lemma 1.8 to be proved later on. The consideration of non-trivial case, i.e. $s > 1$ and $l > s+1$ gives rise to the restriction on $\check{l}^\star$, namely $1 < \check{l}^\star < l-1$. The cases $\check{l}^\star = 1$ or $\check{l}^\star = l-1$ are only inevitable in trivial cases. Otherwise, $\check{l}^\star$ can be successively shifted to satisfy $1 < \check{l}^\star < l-1$ without causing an additional effort in terms of primal time steps needed for the overall reversal.

2nd case: $\check{m}^\star = \check{l}^\star + 1$. Therefore,

$$\begin{gathered}\mathbf{T}_{bin}(l+1, s) - \mathbf{T}_{bin}(l, s) = \check{l}^\star + 1 + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \\ + \mathbf{T}_{bin}(\check{l}^\star + 1, s) - \check{l}^\star - \mathbf{T}_{bin}(l-\check{l}^\star, s-1) - \mathbf{T}_{bin}(\check{l}^\star, s) = \\ = 1 + \mathbf{T}_{bin}(\check{l}^\star + 1, s) - \mathbf{T}_{bin}(\check{l}^\star, s) = \\ = r(s, \check{l}^\star + 1) + 1 = r(s, l+1).\end{gathered} \tag{30}$$

Since $1 < \check{l}^\star < l-1$, we obtain $\mathbf{T}_{bin}(\check{l}^\star+1, s) - \mathbf{T}_{bin}(\check{l}^\star, s) = r(s, \check{l}^\star+1)$ due to the induction assumption. The last equality in (30) is because of Lemma 1.8 to be proved later on.

This proves the equation (26), which implies (25). Therefore, the equivalence between the equation (19) and the equation (22) is shown, which immediately yields the statement of Corollary 1.6.

□

In the following, we prove two lemmas, those results are used throughout the alternative proof of Corollary 1.6.

Lemma 1.7. *Assume that the following equation is true for a fixed s and all $\tilde{l}$ with $\tilde{l} < l$:*

$$\mathbf{T}_{bin}(\tilde{l}+1, s) - \mathbf{T}_{bin}(\tilde{l}, s) = r(s, \tilde{l}+1), \tag{31}$$

with $\mathbf{T}_{bin}(\tilde{l}+1, s)$ and $\mathbf{T}_{bin}(\tilde{l}, s)$ being evaluated using (19). Moreover, assume that an optimal value $\check{l}^\star$ is determined, so that

$$\mathbf{T}_{bin}(l, s) = \check{l}^\star + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star, s). \tag{32}$$

Then, an optimal value $\check{m}^\star$ for $m = l+1$ has two possibilities to choose from, namely $\check{m}^\star = \check{l}^\star$ or $\check{m}^\star = \check{l}^\star + 1$.

Proof. Since, an optimal value $\check{l}^\star$ is determined, so that (32) satisfied, then, the following two relations are valid.

$$\check{l}^\star + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star, s) \leq \check{l}^\star - 1 + \mathbf{T}_{bin}(l-\check{l}^\star+1, s-1) + \mathbf{T}_{bin}(\check{l}^\star - 1, s), \tag{33}$$

for $\check{l}^\star > 1$ and

$$\check{l}^\star + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star, s) \leq \check{l}^\star + 1 + \mathbf{T}_{bin}(l-\check{l}^\star-1, s-1) + \mathbf{T}_{bin}(\check{l}^\star + 1, s), \tag{34}$$

for $\check{l}^\star < l-1$. Since the function $\mathbf{T}_{bin}(l,s)$ is convex, the relations (33) and (34) are equivalent to $\check{l}^\star$ being its global minimum defined by (19). Because of the assumption (31), the equations (33) and (34) are equivalent to

$$r(s, \check{l}^\star) + 1 \leq r(s-1, l-\check{l}^\star+1), \tag{35}$$

for $\check{l}^\star > 1$ and

$$r(s, \check{l}^\star + 1) + 1 \geq r(s-1, l-\check{l}^\star), \tag{36}$$

for $\check{l}^\star < l-1$, respectively. The relations (35) and (36) are true if $\check{l}^\star$ is an optimal value satisfying (32).

In order to prove the assertion of Lemma 1.7, i.e. to prove that an optimal value $\check{m}^\star$ for $m = l+1$ has two possibilities to choose from, namely $\check{m}^\star = \check{l}^\star$ or $\check{m}^\star = \check{l}^\star + 1$, we have to show the following two relations are satisfied.

$$\check{l}^\star + \mathbf{T}_{bin}(l+1-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star, s) \leq \check{l}^\star - 1 + \mathbf{T}_{bin}(l-\check{l}^\star+2, s-1) + \mathbf{T}_{bin}(\check{l}^\star - 1, s), \tag{37}$$

for $\check{l}^\star > 1$ and

$$\check{l}^\star + 1 + \mathbf{T}_{bin}(l-\check{l}^\star, s-1) + \mathbf{T}_{bin}(\check{l}^\star + 1, s) \leq \check{l}^\star + 2 + \mathbf{T}_{bin}(l-\check{l}^\star-1, s-1) + \mathbf{T}_{bin}(\check{l}^\star + 2, s), \tag{38}$$

for $\check{l}^\star < l-1$. (37) and (38) identify $\check{l}^\star$ or $\check{l}^\star + 1$ as a global minimum defined by (19) for $l+1$ time steps. Because of the assumption (31), the equations (37) and (38) are equivalent to the following two relations

$$r(s, \check{l}^\star) + 1 \leq r(s-1, l-\check{l}^\star+2), \tag{39}$$

for $\check{l}^\star > 1$ and

$$r(s, \check{l}^\star + 2) + 1 \geq r(s-1, l-\check{l}^\star), \tag{40}$$

for $\check{l}^\star < l-1$, respectively. Recalling (35) and using the fact that the function $r(s,l)$ is monotone increasing in l, we obtain that $r(s-1, l-\check{l}^\star+1) \leq r(s-1, l-\check{l}^\star+2)$. Therefore, the inequality (39) is true.

Moreover, with the observations above we obtain $r(s, \check{l}^\star + 2) \geq r(s, \check{l}^\star + 1)$. Consequently, the inequality (40) is also satisfied due to (36). Hence, the assertion of Lemma 1.7 is proved.

□

Lemma 1.8. *Consider a reversal schedule* $\mathbf{S}$ *which application to the implementation of a double sweep evolution* $\mathcal{E}_{2\times m}$ *with* $m = l+1$, $l > s+1$, *and* s *being the number of available checkpoints causes a minimal number* $\mathbf{T}_{bin}(m,s)$ *of primal steps, i.e.*

$$\mathbf{T}(\mathbf{S}) = \mathbf{T}_{bin}(m,s) = \min_{0<\check{m}<m} \{\check{m} + \mathbf{T}_{bin}(m-\check{m}, s-1) + \mathbf{T}_{bin}(\check{m}, s)\}. \tag{41}$$

Moreover, assume that the following equation is true for a fixed s and all $\tilde{l}$ with $\tilde{l} < l$ and $l > s + 1$.

$$\mathbf{T}_{bin}(\tilde{l}+1, s) - \mathbf{T}_{bin}(\tilde{l}, s) = r(s, \tilde{l}+1), \tag{42}$$

with $\mathbf{T}_{bin}(\tilde{l}+1, s)$ and $\mathbf{T}_{bin}(\tilde{l}, s)$ evaluated using (19). Furthermore, an optimal value $\check{m}^\star$ is determined, so that

$$\mathbf{T}_{bin}(m, s) = \check{m}^\star + \mathbf{T}_{bin}(m - \check{m}^\star, s-1) + \mathbf{T}_{bin}(\check{m}^\star, s). \tag{43}$$

Then, for the maximal repetition numbers $r_{max}(\mathbf{S})$, $r_{max}(\mathbf{S}^1)$, and $r_{max}(\mathbf{S}^2)$ for the reversal schedules $\mathbf{S}$, $\mathbf{S}^1$, and $\mathbf{S}^2$, respectively, so that $\mathbf{S} = \mathbf{S}^1 \circ \mathbf{S}^2$, $\mathbf{T}(\mathbf{S}^1) = \mathbf{T}_{bin}(\check{m}^\star, s)$, and $\mathbf{T}(\mathbf{S}^2) = \mathbf{T}_{bin}(m - \check{m}^\star, s-1)$, the following relations are satisfied

$$r(s, m) = r(s, \check{m}^\star) + 1 = r(s-1, m - \check{m}^\star), \tag{44}$$

with $r_{max}(\mathbf{S}) = r(s, m)$, $r_{max}(\mathbf{S}^1) = r(s, \check{m}^\star)$, and $r_{max}(\mathbf{S}^2) = r(s-1, m - \check{m}^\star)$.

Proof. Suppose, relations (44) are not satisfied, i.e. $r(s, \check{m}^\star) + 1 > r(s-1, m - \check{m}^\star)$ or $r(s, \check{m}^\star) + 1 < r(s-1, m - \check{m}^\star)$. In the following, consider each case separately.

1st case: Suppose, $r(s, \check{m}^\star) + 1 > r(s-1, m - \check{m}^\star)$. If $\check{m}^\star = 1$, then $1 > r(s-1, m-1) \geq 1$, which implies immediately the contradiction. Thus, we exclude the case $\check{m}^\star = 1$ from the beginning. Therefore, this is equivalent to

$$r(s, \check{m}^\star) + 1 \geq r(s-1, m - \check{m}^\star) + 1 \geq r(s-1, m + 1 - \check{m}^\star). \tag{45}$$

Since an optimal value $\check{m}^\star$ is determined, so that (43) is satisfied, then, the following relation is valid.

$$\begin{aligned} \check{m}^\star + \mathbf{T}_{bin}(m - \check{m}^\star, s-1) + \mathbf{T}_{bin}(\check{m}^\star, s) \leq \check{m}^\star - 1 \quad &+ \quad \mathbf{T}_{bin}(m + 1 - \check{m}^\star, s-1) + \\ &+ \quad \mathbf{T}_{bin}(\check{m}^\star - 1, s). \end{aligned} \tag{46}$$

Because of the assumption (42) and $\check{m}^\star > 1$, the equation (46) is equivalent to

$$r(s, \check{m}^\star) + 1 \leq r(s-1, m + 1 - \check{m}^\star). \tag{47}$$

If the last inequality in (45) and the relation (47) are satisfied strictly, then, they contradict each over. Therefore, the assumption $r(s, \check{m}^\star) + 1 > r(s-1, m - \check{m}^\star)$ is not true. In the other case, i.e. if $r(s, \check{m}^\star) + 1 = r(s-1, m + 1 - \check{m}^\star)$, we can move the optimal value $\check{m}^\star$ towards the left by one without increasing the resulting evaluation cost $\mathbf{T}(\mathbf{S}) = \mathbf{T}_{bin}(m, s)$ due to (46). Applying the considerations above sufficiently many times, we achieve relation (46) and consequently (47) are satisfied strictly, which implies the contradiction between (45) and (47). Thus, we proved that the relation

$$r(s, \check{m}^\star) + 1 > r(s-1, m - \check{m}^\star)$$

is not true.

2nd case: Suppose, $r(s, \check{m}^\star) + 1 < r(s-1, m - \check{m}^\star)$. If $\check{m}^\star = m - 1$, then $0 > 1 + r(s, m-1) \geq 1$, which implies immediately the contradiction. Thus, we exclude the case $\check{m}^\star = m - 1$ from the following considerations. Therefore, this is equivalent to

$$r(s, \check{m}^\star) + 2 \leq r(s-1, m - \check{m}^\star). \tag{48}$$

Since, an optimal value $\check{m}^\star$ is determined, so that (43) is satisfied, then, the following relation is valid.

$$\begin{aligned}(49)\qquad \check{m}^\star + \mathbf{T}_{bin}(m-\check{m}^\star, s-1) + \mathbf{T}_{bin}(\check{m}^\star, s) \le \check{m}^\star + 1 \quad &+ \quad \mathbf{T}_{bin}(m-\check{m}^\star - 1, s-1) \\ &+ \quad \mathbf{T}_{bin}(\check{m}^\star + 1, s).\end{aligned}$$

Because of the assumption (42) and $\check{m}^\star < m-1$, the equation (49) is equivalent to

$$(50)\qquad r(s, \check{m}^\star + 1) + 1 \ge r(s-1, m-\check{m}^\star).$$

Since $1 + r(s, \check{m}^\star) \ge r(s, \check{m}^\star + 1)$, we obtain from (50)

$$(51)\qquad r(s, \check{m}^\star) + 2 \ge r(s-1, m-\check{m}^\star).$$

If the inequalities (48) and (51) are satisfied strictly, then, they contradict each over. Therefore, the assumption $r(s, \check{m}^\star) + 1 < r(s-1, m-\check{m}^\star)$ is not true. In the other case, i.e. if $r(s, \check{m}^\star) + 2 = r(s-1, m-\check{m}^\star)$, we can move the optimal value $\check{m}^\star$ towards the right by one without increasing the resulting evaluation cost $\mathbf{T}(\mathbf{S}) = \mathbf{T}_{bin}(m, s)$ due to (49). Applying the considerations above sufficiently many times, we achieve relation (50) and consequently (51) are satisfied strictly, which implies the contradiction between (49) and (51). Thus, we proved that the relation

$$r(s, \check{m}^\star) + 1 = r(s-1, m-\check{m}^\star)$$

is true.

Furthermore, we have to prove at least one of the relations

$$(52)\qquad r(s,m) = r(s, \check{m}^\star) + 1, \qquad \text{or} \qquad r(s,m) = r(s-1, m-\check{m}^\star).$$

They follow immediately from the decomposition $\mathbf{S} = \mathbf{S}^1 \circ \mathbf{S}^2$. Therefore, due to (13), we obtain

$$\begin{aligned}(53)\qquad & r(s,m) = r_{max}(\mathbf{S}) = r_{max}(\mathbf{S}^1 \circ \mathbf{S}^2) = \\ & \max\left\{r_{max}(\mathbf{S}^1) + 1, r_{max}(\mathbf{S}^2)\right\} = \\ & \max\left\{r(s, \check{m}^\star) + 1, r(s-1, m-\check{m}^\star)\right\} = \\ & = r(s, \check{m}^\star) + 1 = r(s-1, m-\check{m}^\star).\end{aligned}$$

Hence, the assertion of Lemma 1.8 is proved.

□

Corollary 1.6 implies the following conclusion. If a sequence of l time steps is reversed, using a minimal number $\mathbf{T}_{bin}(l, s)$ of forward steps and s checkpoints, each time step F_i, $1 \le i \le l$, can be associated with a unique number k_i, $1 \le k_i \le l$. The repetition number $r(s, k_i) = r_{bin}(s, k_i)$ counts how often the time step F_i is evaluated during the reversal of the step sequence. These relations are illustrated in Figure 9. Obviously, it can occur that there

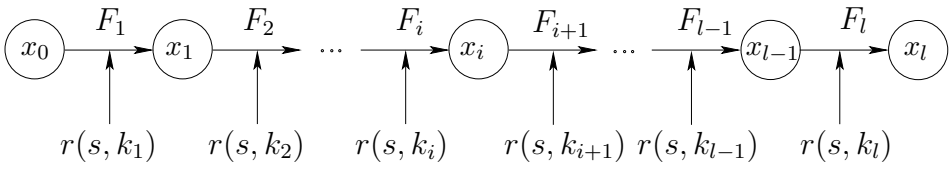

Figure 9: Correspondence between time steps and repetition numbers

are several different steps F_i and consequently different numbers k_i with $r(s,k_i) = j$ for a specified j, $1 \leq j \leq r_{bin}(s,l)$.

The dependence of the minimal evaluation cost $\mathbf{T}_{bin}(l,s)$ on l is shown in Figure 10 for various numbers s of checkpoints. $\mathbf{T}_{bin}(l,s)$ being a function on l for fixed s produces piecewise linear graphs.

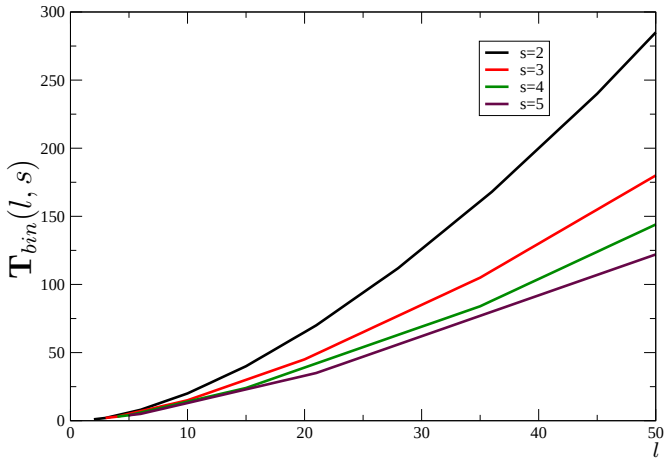

Figure 10: Minimal evaluation cost $\mathbf{T}_{bin}(l,s)$ depending on l and s

The relation (22) gives an explicit dependence for a minimal cost $\mathbf{T}_{bin}(l,s)$ on $r(s,i)$, $i = 1,...,l$. Hence, $r(s,l)$ represents the absolute growth of $\mathbf{T}_{bin}(l,s)$ compared to $\mathbf{T}_{bin}(l-1,s)$. Obviously, relation (22) implies

$$\mathbf{T}_{bin}(l,s)/l \leq r(s,l). \tag{54}$$

Thus, $r(s,l)$ also represents an upper bound for the ratio between the minimal cost $\mathbf{T}_{bin}(l,s)$, needed to reverse l primal steps with up to s available checkpoints, and the overall cost for the primal sweep, i.e. the temporal complexity needed to implement a forward simulation F comprising l time steps.

Figure 11 illustrates level sets of the maximal length $l_{max}(r,s)$ as a function on r and s. Each curve in Figure 11 corresponds to a specified number $l_{max}(r,s)$. On this figure we can also indicate the approximate functional dependence of l as a function on r and s when either of them is constant or both are proportional to each other. For a fixed repetition number r the number s of required checkpoints grows like the length l raised to the reciprocal of the repetition number r. For a fixed number s of available checkpoints the relative reversal cost r grows like the length l raised to the reciprocal of the number s of checkpoints available. In the case $r \sim s$ the relative reversal cost r as well as the number s of checkpoints available grow like the logarithm of the length l.

Thus, if we can arrange that $s = ln(l)$, i.e. if it is possible to allocate space for s checkpoints, then it is possible to find such a reversal schedule $\mathbf{S}$ with $\mathbf{T}(\mathbf{S}) = \mathbf{T}_{bin}(l,s)$ and apply it to the implementation of the double sweep evolution $\mathcal{E}_{2\times l}$. The growth of the temporal complexity, caused by this implementation, compared to the temporal complexity of a simple forward simulation, and counted in the number of primal steps, can be bounded from above by the natural logarithm of a number of time steps, comprised in the forward simulation.

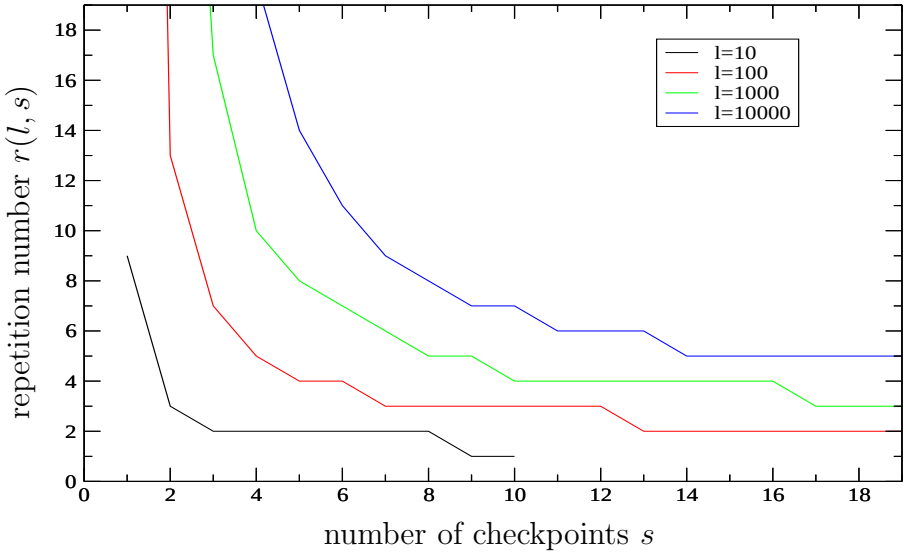

Figure 11: Level sets of maximal length $l_{max}(r,s)$

2 Nested reversal schedules

In the present section we introduce a formal concept for nested checkpointing. Investigation of nested checkpointing is motivated by the problem of reduction of memory requirement needed to implement Newton step of Pantoja type for optimal control problem in ODEs (see [2, 6, 8, 9]). Moreover, nested checkpointing can be applied for the spatial discretization of PDEs with adaptive FE grids.

Nested checkpointing techniques represent an extension for checkpointing considered so far in this paper. Using nested checkpointing we will tackle the problem of implementing triple-sweep evolutions. We discuss its main properties and give a survey of available strategies to construct nested reversal schedules. Finally, we estimate the computational effort, needed to implement triple-sweep evolutions using nested checkpointing techniques.

2.1 Formalism

Consider a multiple sweep evolution $\mathcal{E}_{3\times l}$ which contains three alternative sweeps. Each sweep consists of l consecutive time steps. An example of such an evolution is given in Figure 12. Time steps are shown as horizontal arrows. Their directions denote the information flow. Nodes denote different intermediate states. Each sweep is characterized by one specified direction, i.e. the direction of horizontal arrows within a single sweep. It shows the corresponding information flow between neighboring intermediate states of a single sweep. Note that the information flow within a single sweep has a constant direction. An additional information flow exists between nodes being intermediate states of different successive sweeps. This is shown by vertical lines in Figure 12.

We define intermediate states of the primal, adjoint, and final sweeps by x_i, $\bar{x}_i$, and $\bar{\bar{x}}_i$, $0 \leq i \leq l$, respectively. In the same manner, we identify time steps of various sweeps with F_i,

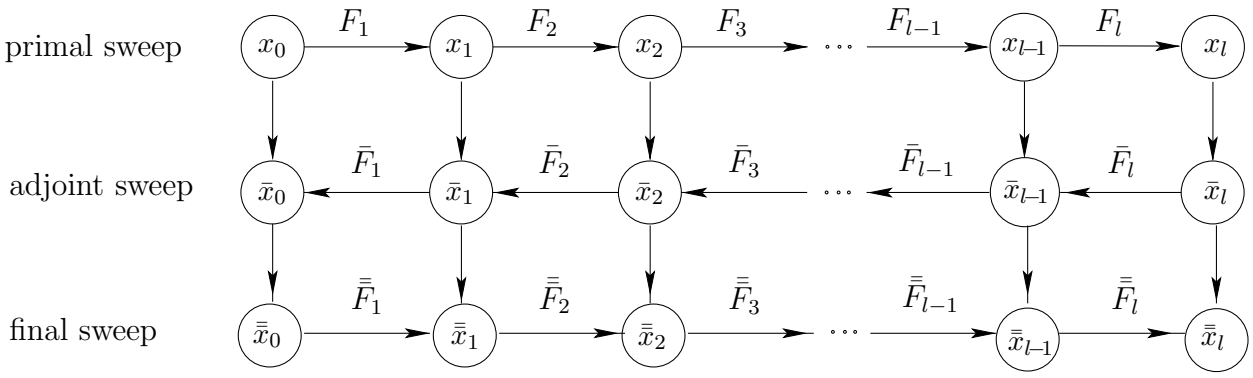

Figure 12: Multiple sweep evolution $\mathcal{E}_{3\times l}$

$\bar{F}_i$, and $\bar{\bar{F}}_i$, $1 \le i \le l$. Then, we obtain

$$x_i = F_i(x_{i-1}), \quad \bar{x}_{i-1} = \bar{F}_i(x_{i-1}, \bar{x}_i), \quad \bar{\bar{x}}_i = \bar{\bar{F}}_i(\bar{x}_i, \bar{\bar{x}}_{i-1}), \quad 1 \le i \le l. \tag{55}$$

We assume that sizes of intermediate states are constant within a single sweep. Therefore, we denote them

$$d \equiv \text{ size } \{x_i\}, \quad \bar{d} \equiv \text{ size } \{\bar{x}_i\}, \quad \bar{\bar{d}} \equiv \text{ size } \{\bar{\bar{x}}_i\}. \tag{56}$$

Moreover, we introduce evaluation costs, i.e. the computational effort for time steps of different sweeps. We assume that within each single sweep we have uniform step costs, i.e. three constants t, $\bar{t}$, and $\bar{\bar{t}}$ exist there, so that

$$t \equiv TIME(F_i), \quad \bar{t} \equiv TIME(\bar{F}_i), \quad \bar{\bar{t}} \equiv TIME(\bar{\bar{F}}_i), \quad 1 \le i \le l. \tag{57}$$

Furthermore, we assume that

$$\bar{d} \gg d \qquad \text{and} \qquad \bar{t} \gg t. \tag{58}$$

Thus, the size $\bar{d}$ of the adjoint state is much larger than the size d of the primal state. Correspondingly, the evaluation of adjoint steps is much more extensive than the evaluation of primal steps. The relations (55) and the assumptions (58) result from the basic properties of Newton/Pantoja/Riccati step, which memory reduced implementation represents a motivation for investigation of nested checkpointing techniques.

The aim is to implement an evolution $\mathcal{E}_{3\times l}$ using nested checkpointing. The problem is how to place different checkpoints in order to implement the evolution $\mathcal{E}_{3\times l}$ most efficiently. We call each possible strategy as **nested reversal schedule** because checkpoints are set and released at two different levels. Since the information to be stored on the primal sweep differs from that needed on the adjoint sweep, we have two classes of checkpoints. Hence, we call the checkpoints **primal** on the primal sweep and **adjoint** on the adjoint sweep. The dimensions of primal and adjoint checkpoints are defined by

$$\text{size(primal)} \;=\; \text{size}(x_i) = d, \qquad i = 0, ..., l, \tag{59}$$

$$\text{size(adjoint)} \;=\; \text{size}(\bar{x}_i) = \bar{d}, \qquad i = 0, ..., l. \tag{60}$$

While the length of steps may vary arbitrarily with respect to the physical time increment they represent, we assume throughout that the total number l of time steps is a priori known.

When this is not the case, an upper bound on l may be used, which results of course in some loss of efficiency. Fully adaptive nested reversal schedules are under development.

Thus, it is obviously not required to store intermediate states of the final sweep as checkpoints since information computed during this sweep is required just for subsequent time steps within this sweep but not for previous sweeps. If only a restricted amount of memory is available, it is convenient to measure its size in terms of the number of adjoint checkpoints which can be accommodated by them. Moreover, it is assumed that one primal and one adjoint checkpoints are available additionally. The initial primal state and the final adjoint state are to be kept in these additional checkpoints throughout the execution of a nested reversal schedule.

Since adjoint checkpoints, i.e. checkpoints of the size $\bar{d}$, have to be stored during the adjoint sweep, we can use available memory on the primal sweep to store primal checkpoints, i.e. checkpoints of the size d, to reduce the total number of evaluated primal steps F_i. A parameter c denotes how many primal checkpoints can be stored in place of a adjoint one. Obviously, the following is valid

$$\text{size(adjoint)} = \bar{d} = c\,\text{size(primal)} = c\,d, \qquad \Rightarrow \qquad c = \left\lfloor \frac{\bar{d}}{d} \right\rfloor . \tag{61}$$

On the adjoint sweep we successively remove primal checkpoints and store adjoint checkpoints instead of them. This has to be done as soon as required memory is available, i.e. as soon as a sufficient number of primal checkpoints is removed. Each nested reversal schedule that successfully implement an evolution $\mathcal{E}_{3\times l}$ with fixed properties using given resources, i.e. memory available for storing a certain number C of adjoint checkpoints, is called an admissible one. Such schedules will be denoted by $\mathbf{S}$ throughout this paper. For the formal definition of an admissible nested reversal schedule $\mathbf{S}$ see [8].

For an admissible nested reversal schedule $\mathbf{S}$ the **repetition numbers** $r_i \equiv r(i)$, $\bar{r}_i \equiv \bar{r}(i)$, and $\bar{\bar{r}}_i \equiv \bar{\bar{r}}(i)$, defined by the functions

$$r, \bar{r}, \bar{\bar{r}} : [1, l] \to \mathbb{N}, \tag{62}$$

count how often the ith primal, the ith adjoint, and the ith final step, is evaluated during the execution of the nested reversal schedule $\mathbf{S}$.

Figure 13 shows one example for an admissible nested reversal schedule $\mathbf{S}$ for an evolution $\mathcal{E}_{3\times 9}$ with the size distribution $\vec{\mathbf{d}}_{\mathbf{3\times 9}} = (d, \bar{d}, \bar{\bar{d}})^\top = (1, 1, 1)^\top$. $\mathcal{E}_{3\times 9}$ contains three sweeps each of which consists of nine time steps. Two adjoint checkpoints are available, i.e. two adjoint states can be stored except for the final adjoint state $\bar{x}_9$ which data is stored into the additional adjoint checkpoint. Moreover, one primal checkpoint can be stored in place of a single adjoint one, i.e. $c = 1$. Furthermore, the initial primal state x_0 is stored into the additional primal checkpoint. In Figure 13 time steps are plotted along the vertical axis. Time required for the implementation of the evolution $\mathcal{E}_{3\times 9}$ measured in number of executed steps is represented by the horizontal axis. Hence, the horizontal axis can be thought of as a computational axis. Each solid primal horizontal line including the horizontal axis itself represents a primal checkpoint, i.e. a checkpoint of the size $d = 1$. Each solid thick horizontal line represents a adjoint checkpoint, i.e. a checkpoint of the size $\bar{d} = 1$. Solid slanted primal lines represent primal steps $F_i,\ i = 1, \ldots, 9$, whereas adjoint steps $\bar{F}_i,\ i = 1, \ldots, 9$, are visualized by dotted slanted lines. Final steps $\bar{\bar{F}}_i,\ i = 1, \ldots, 9$, are drawn by slanted dashed-dotted thick lines.

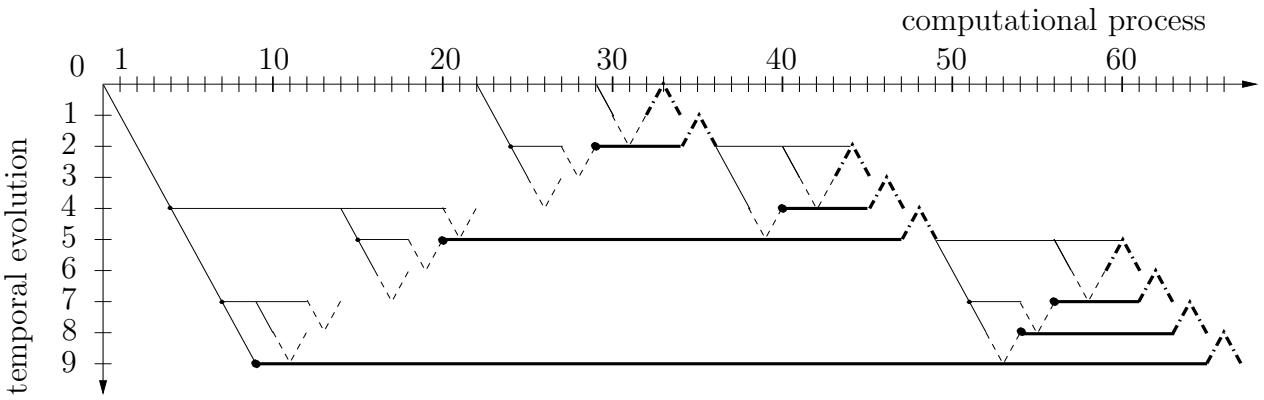

Figure 13: An example for nested reversal schedules $\mathbf{S}$

Nested reversal schedule $\mathbf{S}$ starts with restoring the initial primal state x_0 from the zeroth primal checkpoint. Then, four primal steps F_1, F_2, F_3, and F_4 are performed consecutively. The primal state x_4 is stored into the first primal checkpoint. Now, again, three primal steps F_5, F_6, and F_7 are performed, and the primal state x_7 is stored into the second primal checkpoint. Furthermore, the primal steps F_8 and F_9 are executed and the adjoint state $\bar{x}_9$ is initialized. The adjoint state $\bar{x}_9$ is stored into the additional adjoint checkpoint. The adjoint sweep is started by this action.

Furthermore, data of the primal state x_7 is restored from the second primal checkpoint, the primal step F_8 is performed, and the adjoint states $\bar{x}_8$ and $\bar{x}_7$ can be evaluated. Then, the primal state x_4 is restored from the first primal checkpoint, the primal state x_5 is reevaluated, and its data is stored into the second primal checkpoint. After that, the primal state x_6 is evaluated. Then, the adjoint states $\bar{x}_6$ and $\bar{x}_5$ are evaluated. The adjoint state $\bar{x}_5$ is stored into the first adjoint checkpoint. Then, we go back to the adjoint state $\bar{x}_2$ by reevaluating required primal states and storing the adjoint state $\bar{x}_2$ into the second adjoint checkpoint. Furthermore, one goes back to the adjoint state $\bar{x}_1$ by reevaluating required intermediate states. Consequently, the first final step $\bar{\bar{F}}_1$ is performed. Furthermore, we store the current initial primal state x_1 into the additional primal checkpoint and continue in the same manner in order to execute all other final steps $\bar{\bar{F}}_2, \ldots, \bar{\bar{F}}_9$.

Using the nested reversal schedule $\mathbf{S}$ in Figure 13, it is necessary to perform 23 primal steps F_i, 13 adjoint steps $\bar{F}_i$, and nine final steps $\bar{\bar{F}}_i$. For this reversal schedule $\mathbf{S}$ the corresponding nested repetition profile can be defined by

$$\mathbf{r}(\mathbf{S}) = \begin{pmatrix} r_1 & \ldots & r_9 \\ \bar{r}_1 & \ldots & \bar{r}_9 \\ \bar{\bar{r}}_1 & \ldots & \bar{\bar{r}}_9 \end{pmatrix} = \begin{pmatrix} 3 & 2 & 4 & 2 & 2 & 4 & 2 & 3 & 1 \\ 0 & 1 & 1 & 2 & 2 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}. \tag{63}$$

This profile is illustrated graphically in Figure 14. Each graph in Figure 14 corresponds to a single row of the matrix $\mathbf{r}(\mathbf{S})$ in (63), i.e. to a single sweep of the multiple sweep evolution $\mathcal{E}_{3\times 9}$. x-axis of each graph gives a number i of a corresponding time step F_i, $\bar{F}_i$, $\bar{\bar{F}}_i$, $1 \leq i \leq 9$. y-axis gives a corresponding repetition number r_i, $\bar{r}_i$, or $\bar{\bar{r}}_i$ for a specified time step F_i, $\bar{F}_i$, or $\bar{\bar{F}}_i$, respectively. The vector $\vec{\mathbf{r}}_{max}(\mathbf{S})$ of maximal repetition numbers reaches $\vec{\mathbf{r}}_{max}(\mathbf{S}) = (4, 2, 1)^\top$.

Provided a schedule is admissible, its total runtime complexity can be computed from the

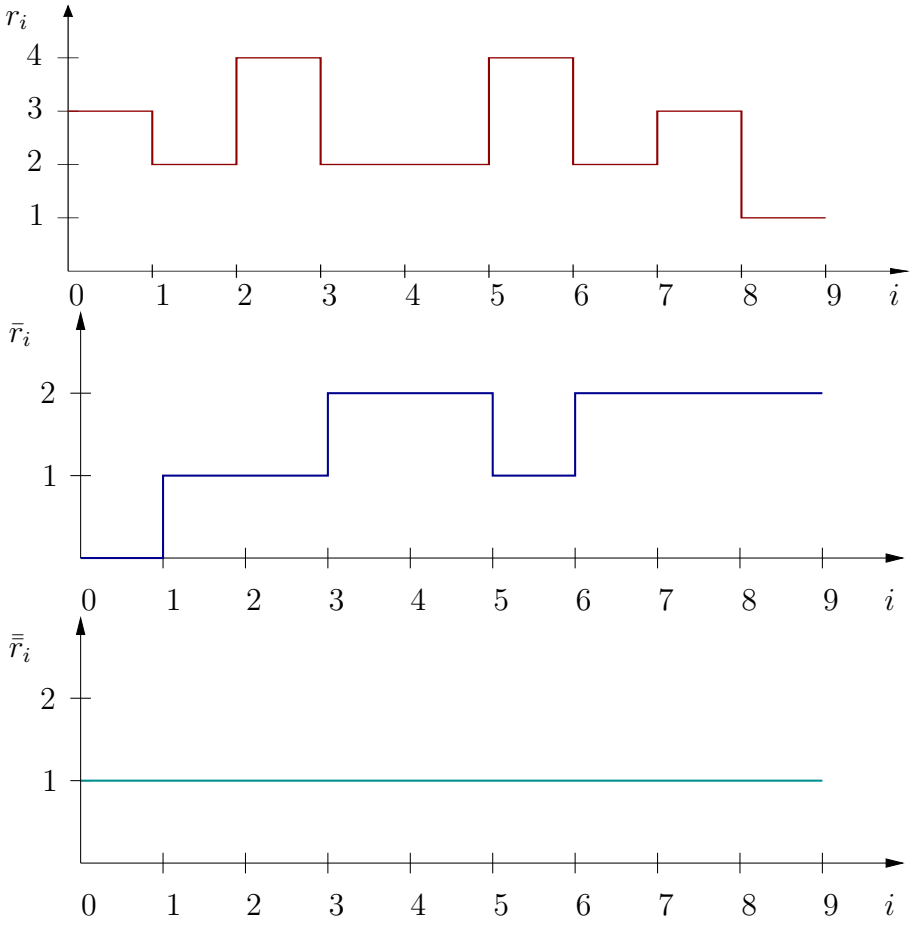

Figure 14: Repetition profiles for $\mathbf{S}$ in Figure 13

additional problem parameters $\vec{\mathbf{t}}_{\mathbf{3\times 1}} = (t, \bar{t}, \bar{\bar{t}})^\top$ by

$$\mathbf{T}(\mathbf{S}) = t \sum_{i=1}^{l} r_i + \bar{t} \sum_{i=1}^{l} \bar{r}_i + \bar{\bar{t}} \sum_{i=1}^{l} \bar{\bar{r}}_i. \tag{64}$$

The temporal complexity of the nested reversal schedule $\mathbf{S}$ in Figure 13 assuming the parameters $\vec{\mathbf{t}}_{\mathbf{3\times 9}} = (1, 5, 1)^\top$ is $\mathbf{T}(\mathbf{S}) = 1 \times 23 + 5 \times 13 + 1 \times 9 = 97$.

The optimal nested reversal schedule from the set of all admissible nested reversal schedules is required to minimize the evaluation cost, i.e. to achieve

$$\mathbf{T}_{min}(l, C) \equiv \min \{\mathbf{T}(\mathbf{S}),\ \mathbf{S} \text{ is admissible}\}. \tag{65}$$

The set of optimal nested reversal schedules is denoted $\mathbf{S}_{min}(l, C)$, so that

$$\mathbf{T}(\mathbf{S}_{min}(l, C)) \equiv \mathbf{T}_{min}(l, C). \tag{66}$$

For the construction of an appropriate optimal nested reversal schedule $\mathbf{S}_{min}(l, C)$ and for the computation of its evaluation cost $\mathbf{T}_{min}(l, C)$ see [8]. In this thesis exhaustive search technique is developed. The nested reversal schedule $\mathbf{S}$ in Figure 13 is an optimal one for the given parameters. This schedule and its minimal evaluation cost are determined using Algorithm 4.1 and routine **optimal** described in [8].

Consider the growth of minimal temporal complexity if the number of steps in the evolution is increased by one. Obviously, the number of final steps to be evaluated through the execution

of $\mathbf{S}_{min}(l, C)$ is also increased by one with respect to $\mathbf{S}_{min}(l-1, C)$. The combination of the adjoint and the final sweep represents a double sweep evolution $\mathcal{E}_{2\times l}$. In Section 1.4 was shown that the minimal number of adjoint steps evaluated through the implementation of $\mathcal{E}_{2\times l}$ compared to the implementation of $\mathcal{E}_{2\times(l-1)}$ is increased by $r(C+1, l)$ with $r(C+1, l) = r_{bin}(C+1, l)$ being repetition number evaluated by (23). The number $(C+1)$ represents the number of C available adjoint checkpoints and the additional adjoint checkpoint, which is considered separately for nested reversal schedules.

In order to evaluate each adjoint step one needs to perform some primal steps. To estimate this number we combine the primal and adjoint sweep to a double sweep evolution $\mathcal{E}_{2\times l}$. Using the similar considerations as above we obtain that the number of primal steps needed to implement $\mathcal{E}_{2\times l}$ is increased by $r(Cc+1, l)$ compared to $\mathcal{E}_{2\times(l-1)}$. Here, the expression $(Cc+1)$ of available checkpoints represents Cc number of primal checkpoints and the additional primal checkpoint, which is also considered separately by nested reversal schedules. Thus, because the overall number of adjoint steps is increased by $r(C+1, l)$, the overall number of primal steps is increased by $r(Cc+1, l)r(C+1, l)$. Therefore, using the definitions (64), (65), and the recent considerations one obtains

$$\mathbf{T}_{min}(l, C) - \mathbf{T}_{min}(l-1, C) \approx t\, r(Cc+1, l)\, r(C+1, l) + \bar{t}\, r(C+1, l) + \bar{\bar{t}}. \tag{67}$$

Generally, the exact equality in (67) is not satisfied, because adjoint and primal checkpoint distributions usually not exactly coincide with optimal checkpoint distribution by single reversal. Nevertheless, the relation (67) gives a very useful approximative estimation of growth behavior for $\mathbf{T}_{min}(l, C)$.

Using investigations concerning temporal complexity of single reversal in Section 1.4, we make the following conclusion. If we can arrange that $C = ln(l)$, i.e. if it is possible to allocate space for C adjoint checkpoints, then it is possible to find such a nested reversal schedule $\mathbf{S}$ with $\mathbf{T}(\mathbf{S}) = \mathbf{T}_{min}(l, C)$ and apply it to the implementation of the triple sweep evolution $\mathcal{E}_{3\times l}$. The growth of the temporal complexity, caused by this implementation, compared to the temporal complexity of a simple forward simulation, can be bounded from above by the second power of the natural logarithm of a number of time steps, comprised in the forward simulation.

2.2 Lower Bound for Minimal Evaluation Cost

The construction of an optimal nested reversal schedule $\mathbf{S}_{min}(l, C)$, as well as the computation of the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ are extremely extensive in run-time and memory consumption. Therefore, it is convenient to determine an estimation for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ before an optimal nested reversal schedule is constructed. A lower bound for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ is denoted $\mathbf{L}_{min}(l, C)$, which can be constructed in the following way.

According to the definition of the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ in (64), the following relation is satisfied

$$\mathbf{T}_{min}(l, C) = \mathbf{T}_f(l, C)t + \mathbf{T}_a(l, C)\bar{t} + l\bar{\bar{t}}, \tag{68}$$

with

$\mathbf{T}_f(l,C)$ $\equiv$ the overall number of primal steps F_i, needed to perform during the execution of $\mathbf{S}_{min}(l,C)$,

$\mathbf{T}_a(l,C)$ $\equiv$ the overall number of adjoint steps $\bar{F}_i$, needed to perform during the execution of $\mathbf{S}_{min}(l,C)$.

Thus, the lower bound $\mathbf{L}_{min}(l,C)$ can be evaluated by

$$\mathbf{L}_{min}(l,C) = \mathbf{L}_f(l,C)t + \mathbf{L}_a(l,C)\bar{t} + l\bar{\bar{t}}, \tag{69}$$

with $\mathbf{L}_f(l,C)$ and $\mathbf{L}_a(l,C)$ being estimations for $\mathbf{T}_f(l,C)$ and $\mathbf{T}_a(l,C)$, respectively, so that the following relations are satisfied

$$\mathbf{L}_f(l,C) \leq \mathbf{T}_f(l,C) \qquad \text{and} \qquad \mathbf{L}_a(l,C) \leq \mathbf{T}_a(l,C). \tag{70}$$

Therefore, according to (69), we obtain the desired relation

$$\mathbf{L}_{min}(l,C) \leq \mathbf{T}_{min}(l,C). \tag{71}$$

Thus, the task of finding the lower bound $\mathbf{L}_{min}(l,C)$ is reduced to the task of evaluating the corresponding estimations $\mathbf{L}_f(l,C)$ and $\mathbf{L}_a(l,C)$.

Firstly, to evaluate the estimation $\mathbf{L}_a(l,C)$, assume that the primal cost t is neglectable small. Practically, this is often the case. Therefore, the primal sweep can be omitted. The evolution $\mathcal{E}_{3\times l}$, consisting of 3 sweeps, is transformed into the evolution $\mathcal{E}_{2\times l}$, consisting of 2 sweeps, namely, the adjoint sweep and the final sweep. The minimal number of adjoint steps, needed to implement the evolution $\mathcal{E}_{2\times l}$, using up to C checkpoints, corresponds to the minimal evaluation cost $\mathbf{T}_{bin}(l,C+1)$. Thus,

$$\mathbf{L}_a(l,C) = \mathbf{T}_{bin}(l,C+1) = \sum_{i=1}^{l} r(C+1,i), \tag{72}$$

with $r(C+1,i) = r_{bin}(C+1,i)$ evaluated by (23). Obviously, (70) is satisfied for $\mathbf{L}_a(l,C)$, if $\mathbf{L}_a(l,C)$ is evaluated by (72).

To evaluate the estimation $\mathbf{L}_f(l,C)$ for the minimal value $\mathbf{T}_f(l,C)$, consider a partition of l primal steps by the partition value $\check{l}$, where the first primal checkpoint is to be stored (see Figure 15). Then, the following relations are satisfied

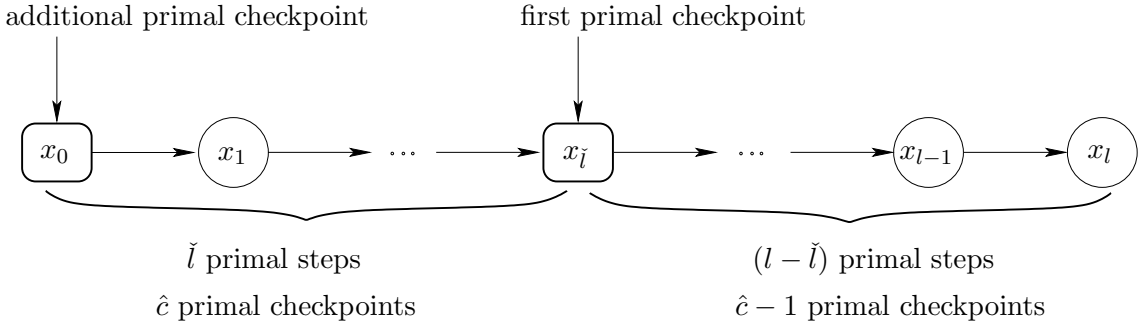

Figure 15: Decomposition of a multiple sweep evolution $\mathcal{E}_{3\times l}$ regarding the primal sweep

$$\begin{aligned} \mathbf{T}_{bin}(l,\hat{c}) = \sum_{i=1}^{l} r_{bin}(\hat{c},i) &\leq \check{l} + \mathbf{T}_{bin}(\check{l},\hat{c}) + \mathbf{T}_{bin}(l-\check{l},\hat{c}-1) \\ &\leq \check{l} + \sum_{i=1}^{\check{l}} r_{bin}(\hat{c},i) + \mathbf{T}_{bin}(l-\check{l},\hat{c}-1). \end{aligned} \tag{73}$$

Note, that if $\check{l}$ is an optimal partitioning number which implies $\mathbf{T}_{bin}(l,\hat{c})$, then the relations in (73) are satisfied with " = " in both cases. Thus, relations in (73) imply

$$\sum_{i=1}^{l} r_{bin}(\hat{c},i) - \sum_{i=1}^{\check{l}} r_{bin}(\hat{c},i) = \sum_{i=\check{l}+1}^{l} r_{bin}(\hat{c},i) \quad \leq \quad \check{l} + \mathbf{T}_{bin}(l-\check{l},\hat{c}-1). \tag{74}$$

During constructing the lower bound $\mathbf{L}_f(l,C)$ the value $\check{l}$ corresponds to the value $\hat{l}_C$, i.e. to the number of adjoint state stored into the Cth adjoint checkpoint. Obviously, this value is initially unknown. Therefore, the sum $\sum_{i=\check{l}+1}^{l} r_{bin}(\hat{c},i)$ is to evaluate exactly in C steps during the construction of $\mathbf{L}_f(l,C)$, i.e.

$$\sum_{i=\check{l}+1}^{l} r_{bin}(\hat{c},i) \quad = \quad \sum_{i=\hat{l}_C+1}^{l} r_{bin}(\hat{c},i) = \sum_{i=\hat{l}_C+1}^{\hat{l}_{C-1}} r_{bin}(\hat{c},i) + \ldots + \sum_{i=\hat{l}_1+1}^{l=\hat{l}_0} r_{bin}(\hat{c},i). \tag{75}$$

Thus, the lower bound $\mathbf{L}_f(l,C) = \mathbf{L}_f(l,C,\hat{c})$ can be recursively constructed by minimizing

$$\mathbf{L}_f(l,C,\hat{c}) \quad = \quad \min_{0<\hat{l}<l} \left\{ \sum_{i=\hat{l}+1}^{l+1} r_{bin}(\hat{c},i) + \mathbf{L}_f^1(l-\hat{l},C,cC+1) + \mathbf{L}_f^1(\hat{l},C-1,\hat{c}) \right\}, \tag{76}$$

with

$$\mathbf{L}_f^1(l,C,\hat{c}) \quad = \quad \min_{0<\hat{l}<l} \left\{ \sum_{i=\hat{l}+1}^{l} r_{bin}(\hat{c},i) + \mathbf{L}_f^1(l-\hat{l},C,cC+1) + \mathbf{L}_f^1(\hat{l},C-1,\hat{c}) \right\}, \tag{77}$$

To evaluate function $\mathbf{L}_f(l,C,\hat{c})$, use Monotonic Approach from [9]. This time, Monotonic Approach has to be applied to the functions $\mathbf{L}_f(l,C,\hat{c})$ and $\mathbf{L}_f^1(l,C,\hat{c})$ instead of to the functions $\mathbf{T}_r(l,C)$ and $\mathbf{T}_r^1(l,C)$. Obviously, Lemma 3.1 and Lemma 3.2 from [9] can be straightforwardly transformed for the functions $\mathbf{L}_f(l,C,\hat{c})$ and $\mathbf{L}_f^1(l,C,\hat{c})$. To find an optimal partitioning point $\hat{l}_\star$ in (76) consider the functions $\mathbf{F}_f(l,C,\hat{c})$ and $\mathbf{F}_f^1(l,C,\hat{c})$, defined by

$$\mathbf{F}_f(l,C,\hat{c}) = \mathbf{L}_f(l,C,\hat{c}) - \mathbf{L}_f(l-1,C,\hat{c}), \qquad \mathbf{F}_f^1(l,C,\hat{c}) = \mathbf{L}_f^1(l,C,\hat{c}) - \mathbf{L}_f^1(l-1,C,\hat{c}). \tag{78}$$

For a specified number l of time steps, number C of available adjoint checkpoints, and a specified parameter c values of the functions $\mathbf{F}_f(l,C,\hat{c})$ and $\mathbf{F}_f^1(l,C,\hat{c})$ can be evaluated recursively. Simultaneously, the lower bound $\mathbf{L}_f(l,C,\hat{c}) = \mathbf{L}_f(l,C)$ for $\mathbf{T}_f(l,C)$ is constructed. The detailed recursive procedure for evaluating $\mathbf{F}_f(l,C,\hat{c})$ and $\mathbf{F}_f^1(l,C,\hat{c})$ is presented in [8].

As a conclusion of the procedure for evaluation $\mathbf{F}_f(l,C,\hat{c})$ and $\mathbf{F}_f^1(l,C,\hat{c})$, the following Algorithm 2.1 is established. This algorithm evaluates the functions $\mathbf{L}_f(l,C,\hat{c})$ and $\mathbf{L}_f^1(l,C,\hat{c})$ for a given number l of time steps, C available checkpoints, and for a specified parameter c.

Therefore, using the estimations $\mathbf{L}_f(l,C) = \mathbf{L}_f(l,C,\hat{c})$ and $\mathbf{L}_a(l,C)$, evaluated by Algorithm 2.1 and (72), respectively, the lower bound $\mathbf{L}_{min}(l,C)$ for the minimal evaluation cost $\mathbf{T}_{min}(l,C)$ can be computed by

$$\mathbf{L}_{min}(l,C) \quad = \quad \mathbf{L}_f(l,C)t + \sum_{i=1}^{l} r_{bin}(C+1,i)\bar{t} + l\bar{\bar{t}}. \tag{79}$$

Algorithm 2.1. Evaluation $\mathbf{L}_f(l,C,\hat{c})$ and $\mathbf{L}^1_f(l,C,\hat{c})$
Input: l - number of time steps
C - number of available checkpoints
c - specified parameter defined by (61)
Output: values of $\hat{m}_\star$, $\hat{m}^1_\star$, $\mathbf{L}_f(m,n,ck+1)$, $\mathbf{L}^1_f(m,n,ck+1)$, $\mathbf{F}_f(m,n,ck+1)$ and $\mathbf{F}^1_f(m,n,ck+1)$) for $0<m\le l$, $0\le n\le C$, and $0\le k\le C$.
For $(m=0;\ m<l;\ m++)$
For $(n=0;\ n\le C;\ n++)$
For $(k=0;\ k\le C;\ k++)$
If $(m+1\le n+1)$
$\hat{g}_\star = m$ for $g=m+1$
$\hat{g}^1_\star = m$ for $g=m+1$
$\mathbf{L}_f(m+1,n,ck+1)=m+1$
$\mathbf{F}_f(m+1,n,ck+1)=1$
$\mathbf{L}^1_f(m+1,n,ck+1)=m$
$\mathbf{F}^1_f(m+1,n,ck+1)=1$
Else
If $(n==0)$
$\mathbf{L}_f(m+1,0,ck+1)=(m+1)+\sum_{i=1}^{m+1}\mathbf{T}_{bin}(i,1)$
$\mathbf{F}_f(m+1,0,ck+1)=\frac{m(m+1)+2}{2}$
$\mathbf{L}^1_f(m+1,0,ck+1)=\sum_{i=1}^{m+1}\mathbf{T}_{bin}(i,1)$
$\mathbf{F}^1_f(m+1,0,ck+1)=\frac{m(m+1)}{2}$
Else
If $\big\{\mathbf{F}^1_f(\hat{m}_\star+1,n-1,ck+1)-r_{bin}(ck+1,\hat{m}_\star+1)<\mathbf{F}^1_f(m+1-\hat{m}_\star,n,cn+1)\big\}$
$\hat{g}_\star=\hat{m}_\star+1$ for $g=m+1$
$\mathbf{F}_f(m+1,n,ck+1)=r_{bin}(ck+1,m+2)-r_{bin}(ck+1,\hat{m}_\star+1)+\mathbf{F}^1_f(\hat{m}_\star+1,n-1,ck+1)$
Else
$\hat{g}_\star=\hat{m}_\star$ for $g=m+1$
$\mathbf{F}_f(m+1,n,ck+1)=r_{bin}(ck+1,m+2)+\mathbf{F}^1_f(m+1-\hat{m}_\star,n,cn+1)$
If $\big\{\mathbf{F}^1_f(\hat{m}^1_\star+1,n-1,ck+1)-r_{bin}(ck+1,\hat{m}^1_\star+1)<\mathbf{F}^1_f(m+1-\hat{m}^1_\star,n,cn+1)\big\}$
$\hat{g}^1_\star=\hat{m}^1_\star+1$ for $g=m+1$
$\mathbf{F}^1_f(m+1,n,ck+1)=r_{bin}(ck+1,m+1)-r_{bin}(ck+1,\hat{m}^1_\star+1)+\mathbf{F}^1_f(\hat{m}^1_\star+1,n-1,ck+1)$
Else
$\hat{g}^1_\star=\hat{m}^1_\star$ for $g=m+1$
$\mathbf{F}^1_f(m+1,n,ck+1)=r_{bin}(ck+1,m+1)+\mathbf{F}^1_f(m+1-\hat{m}^1_\star,n,cn+1)$
$\mathbf{L}_f(m+1,n,ck+1)=\mathbf{L}_f(m,n,ck+1)+\mathbf{F}_f(m+1,n,ck+1)$
$\mathbf{L}^1_f(m+1,n,ck+1)=\mathbf{L}^1_f(m,n,ck+1)+\mathbf{F}^1_f(m+1,n,ck+1)$

Figure 16 illustrates various examples for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ and its lower bound $\mathbf{L}_{min}(l, C)$. This is evaluated by (79) for different values l, C, and c, using the step cost distribution $\vec{\mathbf{t}}_{\mathbf{3}\times\mathbf{1}} = (1, 5, 1)^\top$ and different size distributions $\vec{\mathbf{d}}_{\mathbf{3}\times\mathbf{1}} = (1, c, 1)^\top$. These size distributions depend on parameter c. The horizontal axis denotes the number l of time steps. The vertical axis gives corresponding minimal evaluation costs and its lower bounds. As we can see in Figure 16, the lower bound $\mathbf{L}_{min}(l, C)$, evaluated by the formula

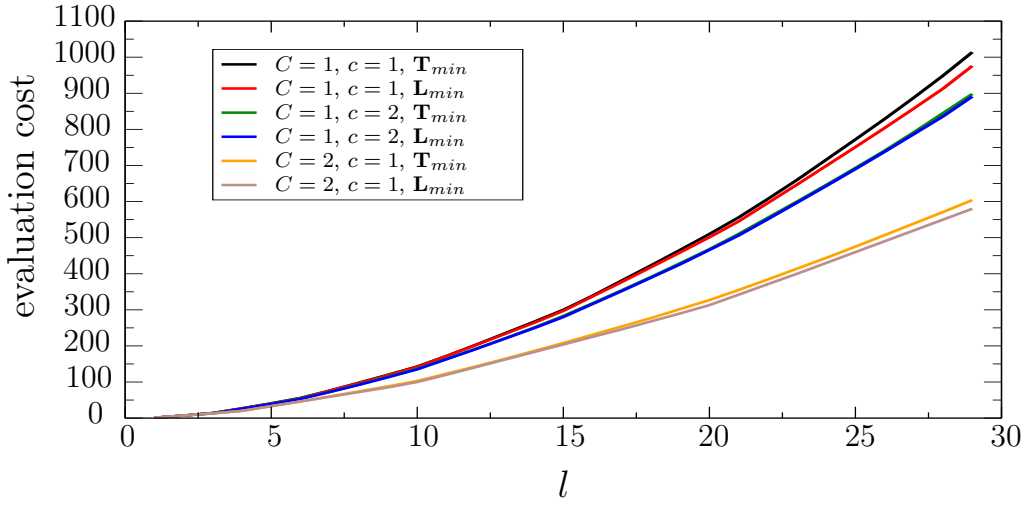

Figure 16: Examples for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ and its lower bound $\mathbf{L}_{min}(l, C)$

(79), lies very close to the minimal evaluation cost $\mathbf{T}_{min}(l, C)$.

Because of the complexity resulting through the construction of optimal nested reversal schedules $\mathbf{S}_{min}(l, C)$ and computation minimal evaluation cost $\mathbf{T}_{min}(l, C)$, it is not possible to construct them for large numbers of l, C, and c. Nevertheless, we can learn more about the quality of its lower bound $\mathbf{L}_{min}(l, C)$, if we compare the temporal complexity $\mathbf{T}_r(l, C)$ of recursively constructed nested reversal schedules $\mathbf{S}_r(l, C)$ and the lower bound $\mathbf{L}_{min}(l, C)$ for the minimal evaluation cost. For detailed construction of nested reversal schedule $\mathbf{S}_r(l, C)$ and for the computation of its evaluation cost $\mathbf{T}_r(l, C)$ see [9]. Temporal complexity $\mathbf{T}_r(l, C)$ represents an upper bound for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$.

Figure 17 shows deviations of the corresponding evaluation cost $\mathbf{T}_r(l, C)$ from the lower bound $\mathbf{L}_{min}(l, C)$ for the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ for different values of l, C, and c, using the step cost distribution $\vec{\mathbf{t}}_{\mathbf{3}\times\mathbf{1}} = (1, 5, 1)^\top$ and different size distributions $\vec{\mathbf{d}}_{\mathbf{3}\times\mathbf{1}} = (1, c, 1)^\top$, which depend on the parameter c. The horizontal axis denotes the number l of time steps. Deviations, evaluated by the formula

$$\frac{(\mathbf{T}_r(l, C) - \mathbf{L}_{min}(l, C))}{\mathbf{L}_{min}(l, C)} \times 100, \tag{80}$$

are plotted along the vertical axis. As we can see in Figure 17, deviations reach up to 20% w.r.t. the lower bound $\mathbf{L}_{min}(l, C)$. Consequently, deviations between the evaluation cost $\mathbf{T}_r(l, C)$ and the minimal evaluation cost $\mathbf{T}_{min}(l, C)$ are even smaller. In our computational experience larger values of c and of ratio $\bar{t}/t$ lead to schedules that are closer to optimal. Therefore, in practice we can expect much smaller deviations. Two last examples with

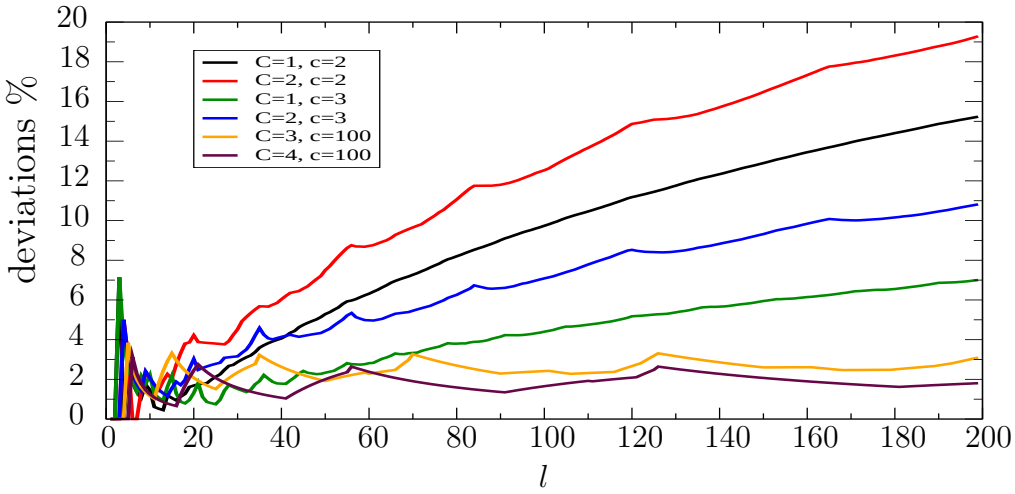

Figure 17: Deviations of $\mathbf{T}_r(l, C)$ from $\mathbf{L}_{min}(l, C)$

$c = 100$ show situations close to realistic. Here, deviations only reach up to 3% w.r.t. the lower bound $\mathbf{L}_{min}(l, C)$.

2.3 Upper Bound for Minimal Evaluation Cost

As explained in [8], the construction of an optimal nested reversal schedule $\mathbf{S}_{min}(l, C)$ by exhaustive search, as well as the computation of the minimal evaluation cost $\mathbf{T}_{min}(l, C)$, are extremely extensive in run-time and memory consumption. Therefore, it is required to construct an appropriate nested reversal schedule $\mathbf{S}_r(l, C)$ (r means here a recursive construction of reversal schedules), so that its evaluation cost $\mathbf{T}_r(l, C)$ differs not too much from the minimal evaluation cost $\mathbf{T}_{min}(l, C)$. Moreover, the construction of a nested reversal schedule $\mathbf{S}_r(l, C)$ has to be carried out using an acceptable memory and run-time requirement. A suitable heuristic, which can be applied to the construction of a nested reversal schedule $\mathbf{S}_r(l, C)$, is developed in [9].

Practically, for the implementation of a triple-sweep evolution $\mathcal{E}_{3\times l}$, a recursively constructed nested reversal schedule $\mathbf{S}_r(l, C)$ is utilized. Therefore, it is very important to estimate the growth of its temporal complexity as it was done for optimal nested reversal schedules in Section 2.1. To this end, it is useful to prove the following assertion.

Theorem 2.2 (**Temporal Complexity Growth**). *The growth of the temporal complexity* $\mathbf{T}_r(l, C)$ *for a recursively constructed nested reversal schedule* $\mathbf{S}_r(l, C)$ *can be estimated as follows*

$$\mathbf{T}_r(l, C) - \mathbf{T}_r(l-1, C) \leq r(C+1, l)^2 t + r(C+1, l)\bar{t} + \bar{\bar{t}}. \tag{81}$$

Proof. To prove the relation (81), it is firstly required to evaluate an upper bound for the evaluation cost $\mathbf{T}_r(l, C)$. To evaluate the upper bound, consider a so called binomial nested reversal schedule $\mathbf{S}^n_{bin}(l, C)$, defined so that during the implementation of $\mathbf{S}^n_{bin}(l, C)$ adjoint and primal checkpoints are placed using the binomial rule (for a detailed explanation of the binomial rule see e.g [7, 13]). For $l = l_{max}(r, C+1)$ being the maximal number of steps,

which can be reversed using $(C+1)$ available checkpoints, so that each step is evaluated up to r times during this reversal, initial primal and adjoint checkpoint distributions of a binomial nested reversal schedule $\mathbf{S}^n_{bin}(l, C)$ are illustrated in Figure 18. Here, the adjoint

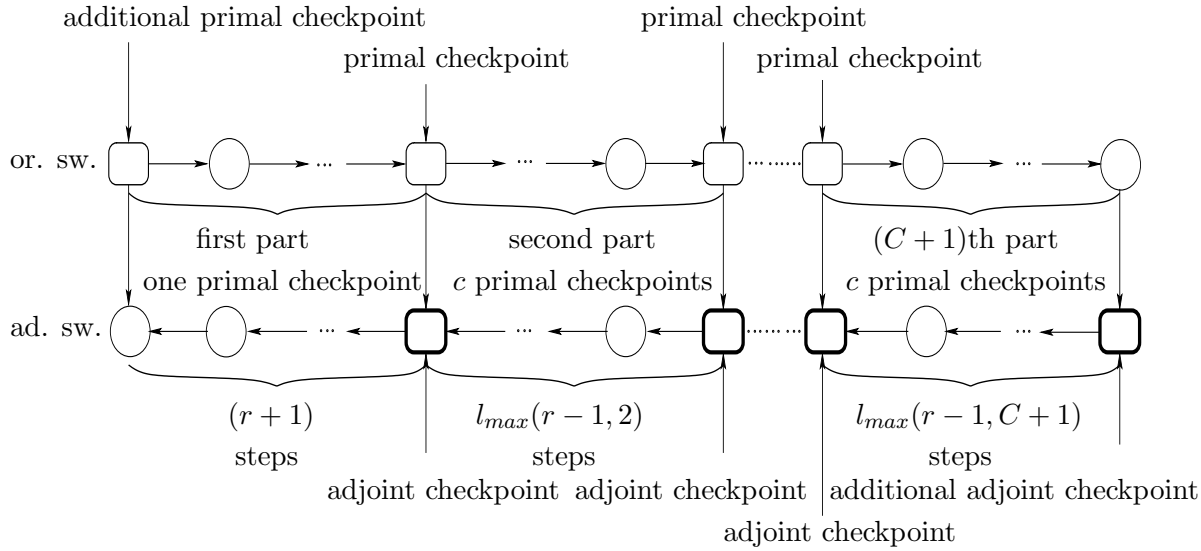

Figure 18: Partition of the primal and the adjoint sweeps of a multiple sweep evolution $\mathcal{E}_{3\times l}$ using the binomial rule with $l = l_{max}(r, C+1)$

sweep is divided into $(C+1)$ parts by C available checkpoints. Each segment has a different length, which is defined by the binomial rule mentioned above. Thus, the length of the ith segment $(i = 2, ..., C+1)$ is evaluated by $l_{max}(r-1, i)$. The length of the first segment is $r + 1 = l_{max}(r, 1)$.

For each segment, except for the first one, c primal checkpoints are available. For the first segment, containing $(r+1)$ steps, only one additional checkpoint is available. Primal checkpoints are also to be set using the binomial rule. This strategy implies a minimal number of primal steps needed to perform in order to reverse each of $(C+1)$ segments.

Due to the construction of nested reversal schedule $\mathbf{S}_r(l, C)$ described in [9], the binomial nested reversal schedule $\mathbf{S}^n_{bin}(l, C)$ is a special case of a recursively constructed nested reversal schedule $\mathbf{S}_r(l, C)$. Therefore, for the evaluation cost $\mathbf{T}^n_{bin}(l, C)$ of the binomial nested reversal schedule $\mathbf{S}^n_{bin}(l, C)$ the following relation is satisfied

$$\mathbf{T}_r(l, C) \leq \mathbf{T}^n_{bin}(l, C). \tag{82}$$

Thus, the evaluation cost $\mathbf{T}^n_{bin}(l, C)$ of the binomial nested reversal schedule $\mathbf{S}^n_{bin}(l, C)$ can be used as an upper bound for the evaluation cost $\mathbf{T}_r(l, C)$. The evaluation cost $\mathbf{T}^n_{bin}(l, C)$ can be computed by

$$\mathbf{T}^n_{bin}(l, C) = \mathbf{T}^f_{bin}(l, C)t + \mathbf{T}^a_{bin}(l, C)\bar{t} + l\bar{\bar{t}}, \tag{83}$$

with $\mathbf{T}^f_{bin}(l, C)$ and $\mathbf{T}^a_{bin}(l, C)$ being the overall number of primal and adjoint steps, respectively, needed to evaluate during the execution of $\mathbf{S}^n_{bin}(l, C)$.

According to the construction of binomial nested reversal schedules, the overall number $\mathbf{T}^a_{bin}(l, C)$ of adjoint steps can be evaluated by

$$\mathbf{T}^a_{bin}(l, C) = \mathbf{T}_{bin}(l, C+1) = \sum_{i=1}^{l} r(C+1, i), \tag{84}$$

with $\mathbf{T}_{bin}(l, C+1)$ being the minimal evaluation cost, needed for the reversal of l steps with up to $(C+1)$ available checkpoints. Thus,

$$\mathbf{T}^a_{bin}(l, C) - \mathbf{T}^a_{bin}(l-1, C) = r(C+1, l). \tag{85}$$

Figure 18 indicates that an upper bound for the number of forward steps, needed to evaluate an adjoint step is

$$\max\left\{r(C+1, l), r(c, m)\right\},$$

with $m = l_{max}\left(r(C+1, l) - 1, C+1\right)$. $r(C+1, l)$ corresponds to the number of forward steps, needed to perform for evaluating an adjoint step from the first part. This part contains $r(l, C+1)+1$ steps, so that only one primal checkpoint is available. $r(m, c)$ corresponds to the number of forward steps, needed to evaluate an adjoint step from the $(C+1)$th part. This part contains $m = l_{max}\left(r(l, C+1) - 1, C+1\right)$ steps, i.e. the maximal number of steps amongst all $(C+1)$ parts, so that c primal checkpoints are available. For a reasonable values of the parameter c, the following property is satisfied

$$\max\left\{r(C+1, l), r(c, m)\right\} = r(C+1, l).$$

Thus, one obtains

$$\mathbf{T}^f_{bin}(l, C) - \mathbf{T}^f_{bin}(l-1, C) \leq r(C+1, l)^2. \tag{86}$$

Then, summarizing (83), (85), and (86), we obtain

$$\mathbf{T}_r(l, C) - \mathbf{T}_r(l-1, C) \leq r(C+1, l)^2 t + r(C+1, l)\bar{t} + \bar{\bar{t}}, \tag{87}$$

which represents exactly the relation (81) to be proved.

□

Theorem 2.2 implies the following powerful conclusion: If we can arrange that $C = ln(l)$, i.e. if it is possible to allocate space for C adjoint checkpoints, and to chose a non-trivial parameter c, then it is possible to construct a nested reversal schedule $\mathbf{S}_r(l, C)$ and apply it to the implementation of the triple sweep evolution $\mathcal{E}_{3\times l}$ with the following property. The growth of the temporal complexity, caused by this implementation, compared to the temporal complexity of a simple forward simulation, can be bounded from above by the second power of the natural logarithm of a number of time steps, comprised in the forward simulation.

3 Conclusion and Outlook

In Table 1 we summarize the temporal complexity results for single and double reversals. In this table, Forward Space and Forward Time represent spatial and temporal complexity of a forward simulation. These complexities are given by a specified constant defining the size of an primal state and lt, respectively, with l being the number of time steps and t being the temporal complexity of the primal step.

Reversal Space in Table 1 denotes the spatial complexity needed for a single or a double reversal using the checkpointing techniques. This complexity is given by

$$\text{Reversal Space} = s \text{ size(primal state)}$$

Single Reversal (Gradient) $s = \ln l$	Double Reversal (Newton Step) $C = \ln l$
$\frac{\text{Reversal Space}}{\text{Forward Space}} = \mathcal{O}(\ln l)$	$\frac{\text{Reversal Space}}{\text{Forward Space}} = \mathcal{O}(\ln^2 l)$
$\frac{\text{Reversal Time}}{\text{Forward Time}} = \mathcal{O}(\ln l)$	$\frac{\text{Reversal Time}}{\text{Forward Time}} = \mathcal{O}(\ln^2 l)$

Table 1: Temporal and spatial complexity for single and double reversal

for a single reversal and

$$\text{Reversal Space} = (Cc + c + 1)\ \text{size(primal state)}$$

in a double reversal case. If C is a number of available adjoint checkpoints, the expression $(Cc + c + 1)$ gives a number of correspondingly available primal checkpoints, incorporating primal and adjoint additional checkpoints.

The Reversal Time in Table 1 denotes the temporal complexity needed to implement a single or a double reversal using the checkpointing techniques. In this table the relations for Reversal Space (Time) and Forward Space (Time) are given for the case if $s = \ln l$, $C = \ln l$, and the parameter c has non-trivial value. Thus, if the memory requirement can be arranged polylogarithmically, then it is possible to arrange that the temporal complexity of single and double reversals grows as the first and second power, respectively, of the natural logarithm of number of time steps.

The investigations above can presumably be generalized for the case of a multiple-sweep evolution $\mathcal{E}_{m\times l}$, with $m \geq 2$ being the number of alternative sweeps. If the memory requirement can be arranged polylogarithmically, then it is possible to arrange that the temporal complexity, needed for the implementation of the multiple-sweep evolution $\mathcal{E}_{m\times l}$, compared to the temporal complexity of a simple forward simulation, grows as the $(m-1)$st power of the natural logarithm of number l of time steps. The proof of this statement is a subject of future investigations.

References

[1] M. Berz, C. Bischof, G. Corliss, and A. Griewank, editors. *Computational Differentiation: Techniques, Applications, and Tools*. SIAM Philadelphia, 1996.

[2] B. Christianson. Cheap Newton steps for optimal control problems: Automatic differentiation and pantoja algorithm. *Optimization Methods and Software*, 10:729–743, 1999.

[3] Y.G. Evtushenko. Automatic differentiation viewed from optimal control. In A. Griewank and G. Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pages 25 – 30. SIAM Philadelphia, 1991.

[4] A. Griewank. *Evaluating derivatives. Principles and techniques of algorithmic differentiation.* SIAM, Philadelphia, 2000.

[5] A. Griewank and A. Walther. Treeverse: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation. *Tech. Report IOKOMO-04-1997, Techn. Univ. Dresden*, 1997.

[6] J. Pantoja. Differential dynamic programming and Newton's method. *International Journal on Control*, 47:1539–1553, 1988.

[7] J. Sternberg. Adaptive Umkehrschemata für Schrittfolgen mit nicht-uniformen Kosten. *Diploma thesis, Institute of Scientific Computing, TU Dresden*, 2002.

[8] J. Sternberg. Reduction of storage requirement by checkpointing for time-dependent optimal control problems. *PhD Thesis, Institute of Scientific Computing, TU Dresden*, 2005.

[9] J. Sternberg and A. Griewank. Nested memory-reduced procedure for solution of optimal control problems using pantoja/riccati method. *In preparation*, 2006.

[10] Y.M. Volin and G.M. Ostrovskii. Automatic computation of derivatives with use of the multilevel differentiation technique. *Computers and Mathematics with Applications*, 11:1099–1114, 1985.

[11] A. Walther. *Program Reversal Schedules for Single- and Multi-processor Machines.* PhD thesis, Institute of Scientific Computing, TU Dresden, 1999.

[12] A. Walther. Program reversals for evolutions with non-uniform step costs. *Acta Informatica*, 40:235–263, 2004.

[13] A. Walther and A. Griewank. Applying the checkpointing routine treeverse to discretizations of burgers' equation. In H.-J. Bungartz, F. Durst, and C. Zenger, editors, *High Performance Scientific and Engineering Computing*, volume 8 of *Lecture Notes in Computational Science and Engineering*, pages 13–24, Berlin/Heidelberg, 1999. Springer.