Running Head: Counterbalancing carryover effects

Counterbalancing for serial order carryover effects in experimental condition orders

Joseph L. Brooks

Institute of Cognitive Neuroscience, University College London

Correspondence to:

Joseph L Brooks

Institute of Cognitive Neuroscience

University College London

17 Queen Square

London WC1N 3AR

United Kingdom

joseph.brooks@ucl.ac.uk

Phone: +44 20 7679 1123

Fax: +44 20 7916 8517

Abstract

Reactions of neural, psychological, and social systems are rarely, if ever, independent of previous inputs and states. The potential for serial order carryover effects from one condition to the next in a sequence of experimental trials makes counterbalancing of condition order an essential part of experimental design. Here, a method is proposed for generating counterbalanced sequences for repeated-measures designs including those with multiple observations of each condition on one participant and self-adjacencies of conditions. Condition ordering is reframed as a graph theory problem. Experimental conditions are represented as vertices in a graph and directed edges between them represent temporal relationships between conditions. A counterbalanced trial order results from traversing an Euler circuit through such a graph in which each edge is traversed exactly once. This method can be generalized to counterbalance for higher-order serial order carryover effects as well as to create intentional serial order biases. Modern graph theory provides tools for finding other types of paths through such graph representations, providing a tool for generating experimental condition sequences with useful properties.

Neural systems, human and otherwise, are characterized by a continuous stream of behavior in which the current state is dependent not only on current inputs but also on previous states and inputs. The history of experimental psychology is replete with examples. The speed with which a participant names a picture can depend on the preceding picture (e.g, Bartram, 1974; Biederman & Cooper, 1991a; 1991b; Goldstein, 1958), a behavioral effect known as priming. Prolonged viewing of a visual stimulus can cause adaptation to its color, orientation, or other features and lead to subsequent aftereffects (e.g., Blakemore & Campbell, 1969; Blakemore & Nachmias, 1971; Bradley, Switkes, & De Valois, 1988). The hemodynamic response (during functional magnetic resonance imaging, FMRI) to a stimulus is attenuated when it has been preceded by a similar stimulus (Grill-Spector & Malach, 2001; Haushofer, Baker, Livingstone, & Kanwisher, 2008; Henson & Rugg, 2003; Vuilleumier, Henson, Driver, & Dolan, 2002). The brain's response to transcranial magnetic stimulation (TMS) also depends on previous visual stimulation (e.g., Pasley, Allen, & Freeman, 2009; Silvanto, Muggleton, & Walsh, 2008). Similar situations can arise in questionnaire assessments used in social psychology and clinical contexts. For instance, survey results can vary depending on the order of the questions (e.g., Faulkner & Cogan, 1990; Gama, Correia, & Lunet, 2009) and election results can be affected by which candidates precede which on the ballot (e.g. Miller & Krosnick, 1998). All of these are examples of *serial order carryover effects*, i.e. the response in one experimental trial in a sequence depends on the preceding trial(s) in that sequence. Because learning and adaptation are central to the functioning of living things, carryover effects are ubiquitous within all domains of psychology and are a standard topic covered in courses and textbooks on experimental design (e.g. see Elmes, Kantowitz, & Roediger, 1999).

In some of the examples given above, serial order carryover effects have been harnessed to give insights into the functioning of neural systems. For instance, FMRI adaptation has been used as a measure of category specificity of brain regions (e.g., Fang, Boyaci, & Kersten, 2009; Gilaie-Dotan, Gelbard-Sagiv, & Malach, 2010) and priming has been widely used to test cognitive models (e.g., Baylis & Cale, 2001; Esterman et al., 2002; Làdavas, Paladini, & Cubelli, 1993). However, serial order carryover effects from one trial to the next can also represent a significant nuisance effect and contribute variance to measurements of effects of interest (e.g., Finney & Outhwaite, 1955; Williams, 1949; Williams, 1952). For instance, serial order carryover effects pose a particular problem in studies concerning taste of food and wine and this issue has received significant attention in that domain (e.g., Duriera, Monoda, & Bruetschy, 1997; Muir & Hunter, 1992; Schlich, 1993; Wakeling & MacFie, 1995). Such carryover effects may account for the common practice of cleansing the palate between courses in fine dining or wine tastings. In *repeated measures* experimental designs (also commonly known as *within-subjects* or *cross-over* designs), in which multiple experimental conditions are presented to each participant in a serial order, serial order carryover effects can be particularly problematic. At worst, if serial order carryover effects exist within a repeated measures design and are not effectively counterbalanced, then they could become confounded with the experimental effects. For instance, Tables 1 and 2 show different sequences of conditions A and B with 10 measures of each condition presented to a given participant. The score in condition A depends on the preceding trial. It is 100 when preceded by another trial of condition A but it is 10 when it is preceded by condition B. Table 1 shows an example sequence of trials in which serial order carryover effects from the immediately preceding trial have been completely counterbalanced, i.e. a *serially balanced* sequence in which each condition is preceded by the other conditions (including itself) equally often. Notice that there is a small difference between the means for conditions A and B as should be the case if serial order effects are to be ignored. In Table 2 however, condition A is preceded more often by itself than by condition B. In this case, because the score for A is larger when preceded by itself than when preceded by B, the average for condition A is inflated by the serial order carryover effect. Unless condition A is preceded equally often by itself and condition B, the direct effect of A will be confounded with the effect of condition order. Serial order effects could be separated from experimental effects post hoc, of course. But if care is not taken to counterbalance against them or assess the effects of serial order afterwards, carryover effects can potentially affect the internal validity of the experiment. In addition to avoiding confounding effects, serially balanced sequences may have applications in domains such as optimal designs in event-related FMRI experiments (Aguirre, 2007; Buracas & Boynton, 2002). In these domains, such sequences have been used to increase the efficiency of estimating the hemodynamic response in FMRI and to increase sensitivity in FMRI adaptation experiments (fMRIa). fMRIa assumes that a population of neurons that responds to a particular feature will adapt (i.e. respond with lower amplitude) to repeated presentations of that feature (Henson & Rugg, 2003; Krekelberg, Boynton, & van Wezel, 2006). For instance, a population of neurons that responds selectively based on facial identity responds with lower amplitude to a face stimulus that was preceded by the same picture than to a face stimulus preceded by a different identity picture (Grill-Spector & Malach, 2001). This adaptation response is useful, for example, in identifying the feature tuning characteristics of neurons. The fMRIa method relies critically on carefully characterizing carryover effects and some have suggested that this is best done with serially balanced sequences (Aguirre, 2007; Buracas & Boynton, 2002).

First-order serial order carryover effects, i.e. those which arise from the immediately preceding trial in a sequence, are the most obvious. However, serial order carryover effects can occur from any number of preceding trials. These higher-order effects can be termed *n^{th}-order* serial order carryover effects where *n* corresponds to the number of preceding trials considered. For instance, *2^{nd}-order* effects would include carryover from the preceding two trials. For each occurrence of conditions A and B in Tables 1 and 2, there are four possible preceding pairs: AA, AB, BA, and BB. Balancing of second order effects in this experiment would require that the 8 possible triplets (AAA, ABA, BAA, BBA, AAB, ABB, BAB, BBB) all occur equally often in the sequence. Higher-order effects can become combinatorially intractable quite quickly but they could theoretically have effects and thus are worth consideration in cases where carryover effects are known to be strong (see example in n^{th}-order counterbalancing section below).

Counterbalancing Techniques

Experiments in cognitive psychology, psychophysics, and cognitive neuroscience (including functional imaging experiments) commonly consist of a sequence of short trials. Each trial represents a particular condition and each condition is commonly repeated many times on different trials. After the experiment, a score is computed for each condition by computing a statistic (e.g., mean or median) across all of the trials within a condition for that participant. The procedure is then repeated (usually with a different sequence) in other participants. Inferential statistics are based on the pattern of scores averaged across all participants.

Experimenters routinely vary the order of conditions within these sequences of experimental trials such that they are not, on average, confounded with condition effects, a practice called *counterbalancing*. The most widely used counterbalancing methods are probably randomization and Latin square designs (e.g. see Elmes et al., 1999).

Randomization of trial order is relatively easy to implement and can be done independently for each participant. The assumption is that any effects of position in the sequence as well as effects due to serial order carryover between adjacent trials will be balanced, on average, across the different randomizations used for different participants in the experiment. However, the random order for any given participant is not guaranteed to be balanced. In fact, first-order serial order carryover effects within a given sequence are *guaranteed to be unbalanced* unless the number of trials in the sequence equals $ac^2 +1$ where c conditions appear equally often in the sequence (except one additional instance of one condition) and a is an integer number of repetitions of each ordered pair in the sequence (which can be used to vary the total number of measurements per condition). The effectiveness of randomization at counterbalancing serial order carryover effects can be assessed post hoc, of course, but it may be difficult or impossible to isolate and remove carryover effects if they are present. Randomization is flexible, however, and can be used to generate sequences of arbitrary length with any number of repetitions of each condition.

Balanced Latin square designs (Bradley, 1958; Edwards, 1951; Elmes et al., 1999) can be used to create a set of sequences which when used together in an experiment (across blocks or participants) will ensure that first order serial carryover effects are balanced although this excludes condition self-adjacencies (e.g. condition A followed by another instance of A). Table 3 shows an example of a balanced Latin square in which there are four conditions. Each row of the Latin square corresponds to the sequence of conditions seen by one participant (or during one block). The sequence for any given participant (or block) is not serially ba1anced. But across all of the participants (or blocks), each condition appears in each serial position equally often and is preceded by every other condition equally often (although not by itself). However, balanced Latin squares do not apply naturally to experiments in which each condition is measured multiple times for each participant. It may be possible to overcome this situation by instead treating rows of the Latin square as sequential blocks of trials to be tested on one participant, however. Also, as mentioned above, balanced Latin squares also do not include any instances of a given condition following itself.

There are methods for creating serially balanced sequences (Finney & Outhwaite, 1955; Nonyane & Theobald, 2007; Sampford, 1957; Williams, 1949; Williams, 1952) which also include multiple replications of each condition for a single participant. An example of a *type 1, index 1* (Finney & Outhwaite, 1955, 1956) sequence is, 1 123456 613254 415263 351462 243165 536421, where the index refers to the number of repetitions of each ordered pair in the sequence. *Type 2* sequences, which do not include instances of condition self-adjacencies, are also possible. Notice that these sequences have a block structure to them with the entire set of conditions, *n*, repeating every *n* trials. These sequences thus balance for serial carryover effects as well as approximate position in the sequence, ensuring that each condition appears regularly throughout the entire length of the sequence. This could be a useful property but it does mean that condition self-adjacencies (e.g., 1 1, 6 6) are fixed to appear only at certain positions, e.g. every six trials in the example above. If condition self-adjacencies are salient events then this could be problematic. Another limitation of these sequences is that, unfortunately, they do not exist for designs with certain numbers of conditions although this may only be the case for 3, 4, and 5 conditions.

M-sequences (Golomb, 1967) have recently been used to generate trial sequences for event-related FMRI (Buracas & Boynton, 2002; Fize et al., 2000), electroencephalography (Baseler, Sutter, Klein, & Carney, 1994; Fortune, Demirel, & Bui, 2009; Li, Bin, Hong, & Gao, 2010), and neurophysiology paradigms (Bernadette & Victor, 1994). These sequences have shown near optimal efficiency in estimating the hemodynamic response in FMRI experiments (Buracas & Boynton, 2002), for instance. M-sequences allow higher-order counterbalancing and have no predictable structure (Aguirre, 2007). However, m-sequences are only available for designs in which the number of conditions, *c*, is equal to the integer power of prime numbers which in practice are 2, 3, or 5. This means that there are m-sequences for *c* = 2, 3 ,4, 5, 8, 9, 16, 25, 27, and so on, but not for 6, 7, 10, 11, etc. More importantly, though, m-sequences are only approximately balanced for serial order carryover effects.

This paper presents a method for generating trial and block orders that are counterbalanced for serial order carryover effects. In other words, each condition is preceded *exactly* equally often by every experimental condition including itself. Importantly, unlike some of the previous methods described above, this method is appropriate for experimental designs involving any number of conditions. Furthermore it can deal with multiple measurements of each condition within a single sequence. The method naturally adapts to allow counterbalancing for higher order serial carryover effects. The proposed framework also provides a general basis for generating other types of sequences that may have useful properties.

Background: Graph Theory

The problem of generating serially balanced sequences can be reformulated in terms of graph theory, the study of mathematical structures used to model pair-wise relations between objects (for an introduction see West, 2001). Graphs are comprised of two types of components, *vertices* and *edges*. Vertices can represent anything in the world, even abstract notions, and edges indicate relationships between them. For example, a graph can be used to represent flights between airports. In Figure 1A, each airport is represented by a vertex in the graph and non-stop flights between the airports are represented by edges between them. The relationship between two vertices can be given a quantitative dimension by assigning the corresponding edge a weight which, for example, may correspond to the flight duration between airports (as in Figure 1A) or the price of a flight between them. Edges can also have directionality. In the directed-graph (digraph) shown in Figure 1A, an edge can only be traversed in the indicated direction. Different directions of travel between two airports are shown as separate edges with different weights because the flight duration is not symmetric. Flying one direction may be slower than flying the opposite direction due to headwinds and tailwinds, for instance. With these simple components graphs can be used to represent a wide array of systems.

The utility of graph representations derives primarily from the large body of knowledge about their mathematical properties, allowing one to reach useful conclusions about relationships between vertices of the graph and, therefore, the represented items. For instance, determining the shortest trip from London to Sydney in Figure 1A requires finding the *path* between the two airports that has the smallest weighted sum of the traversed edges. In this case, there are only two paths through the graph from London to Sydney, and one can simply compute, by hand, that the shortest path from London to Sydney is via Hong Kong. For more complex graphs with many vertices, graph theory provides algorithms for computing various types of paths through graphs and thus can solve a large number of complex optimization problems in the represented domain. Graph representations have been used previously within the domain of experimental design (see

Street & Street, 1987), for instance, to estimate the efficiency of block designs (Paterson, 1983) and to generate tone sequences in which all possible sequential four-tone combinations of nine tone frequencies occur only once (Brimijoin & O'Neill, 2010).

First-Order Counterbalancing

Generating serially balanced sequences for first-order carryover effects can be given a straightforward graph representation. Each condition in the experiment can be represented by a separate vertex in the graph. Because we are interested in the temporal relationships between different conditions (i.e., when in time they are presented), directed edges between vertices will represent the relative timing of different conditions. For instance, traversing a directed edge from the condition A vertex to the condition B vertex in Figure 1B indicates that condition A directly precedes condition B. The ultimate goal is to construct an order of conditions in which each condition is preceded by every condition (including itself) equally often. This goal can be realized through the connectivity of the graph by connecting, in both directions, each vertex to every other vertex in the graph, including a recurrent connection with itself. An example of such a fully-connected, directed graph for four conditions is shown in Figure 1B. This graph represents a system in which the temporal relationships between conditions are perfectly counterbalanced.

To generate a serially balanced sequence from a graph like that in Figure 1B one must find a continuous path through the graph in which each edge is traversed exactly once, thereby representing each temporal relationship equally often in the resulting condition order. Such paths through a graph are called *Euler paths* (or Euler trails) after the mathematician Leonhard Euler (Biggs, Lloyd, & Wilson, 1986; Euler, 1736) who first described them. The path is called an

Euler circuit if it starts and stops at the same vertex. Euler circuits exist in a directed graph (such as the one that we have constructed in Figure 1B) *if and only if* every vertex has equal in and out *degree* (West, 2001). In other words, the number of edges leaving and entering each vertex must be equal. This criterion will always be met for graphs representing our serially balanced condition orders because for every edge out to another vertex, there is always one, and only one, coming back from that vertex and vice versa. A well-known algorithm for finding Euler circuits is Fleury's algorithm (Fleury, 1883; McHugh, 1989). However, several more efficient and easily-programmable algorithms are available. Hierholzer's Algorithm (Fleischner, 1991; Hierholzer, 1873; McHugh, 1989) is shown in Table 4 and described in the example given below. It is easily understandable and efficient. A different algorithm (Kandel, Matias, Unger, & Winkler, 1996) has been proven to draw uniformly (i.e. with equal probability) from the set of all possible Euler circuits through a graph. Because of this, Kandel's algorithm is preferable for generating sequences in most situations. It is possible that Hierholzer's algorithm also draws uniformly from the set of all circuits but, to my knowledge, this has not been proven. Kandel's algorithm is described in Table 5 and implemented in the computer programs provided in the Supplemental Materials.

To illustrate how Euler paths can be constructed, Figure 2 shows an example of applying Hierholzer's Algorithm to a graph with 3 vertices. First, in step 1, a starting vertex, A in this case, is randomly chosen. Steps 2-3 are then repeated several times. We randomly choose an edge leading out of vertex A, from A to A (along the cyclical edge to itself) in this case (edge labeled with *1* in the graph for Steps 2-3 in Figure 2). We mark this edge as traversed and then choose randomly another edge out of vertex A. This time we traverse the edge to vertex B (edge labeled as *2*) and mark this edge as traversed so that it cannot be used again. Following the same procedure we go back to A, then to C, and finally back to A. At this point, there are no further edges that leave vertex A. This means that we have completed a trail, *T*. *T* comprises the following sequence: A A B A C A. There are still edges left in the graph and therefore according to step 3 we continue on. We now move to step 4 in which we choose a new starting vertex which must have been visited in trail *T* created above. All of the vertices had been visited in *T* so we randomly choose vertex B. We then follow steps 5-6 to randomly traverse an edge out of B toward vertex C (edge labeled 1 in graph of Steps 5-6). We continue this process which takes us again to C (along the recursive circular edge), and then back to B and finishes with the cycle that B has with itself. There are no more edges to exit from vertex B which means we have finished this trail. This new trail, *T'*, is B C C B B. Following Step 7 we now replace an instance of vertex B in *T* with trail *T'*. This results in the new serially balanced sequence *T*: A A B C C B B A C A. This algorithm was also used to generate the sequence that is shown in Table 1. The alternative Kandel algorithm is described in Table 5.

For a given number of conditions, there are multiple different trial sequences that can be obtained using the algorithms described above. For an Eulerian directed graph (as is used here to represent the conditions and their temporal relationships) the total number of Eulerian circuits through that graph can be calculated according to the BEST theorem (Aardenne-Ehrenfest & de Bruijn, 1951; Creed & Cryan, 2010; Tutte & Smith, 1941). Using this result, the total number of sequences can be calculated as described in detail in the Appendix. For $c = 2$, there are just 4 possible sequences. This number grows very quickly as the number of conditions, c, increases; 216 (c=3), 331776 (c=4), 2.48 x 10^{10} (c=5), 1.39 x 10^{17} (c=6), 8.26 x 10^{25} (c=7), and so on. For many experiments, the number of possible sequences is very large.

Given the large number of possible sequences for most experiments, an important question is whether each generated sequence is equally useful in practice. Answering this question will, of course, depend on factors particular to each experiment. For instance, some experiments may require careful balance of the positions of conditions in the whole sequence, e.g. making sure that condition 1 appears just as often near the beginning of the sequence as it does near the end. For example, this is important for avoiding a confounding influence of practice effects that may arise over the course of the experiment (e.g. Pashler & Baylis, 1991a; 1991b). Methods have been proposed for assessing the balance of conditions across whole sequence (e.g. Sohn, Bricker, Simon, & Hsieh, 1997). If balance of condition positions over the entire sequence is important, then such methods can be used to select amongst the counterbalanced sequences generated by the algorithm described in this paper. Kandel et al. (1996) proved that their algorithm (which is the one used to generate sequences in the programs provided in the Supplemental Materials) generates sequences which are selected uniformly from the set of all Euler paths through a graph. Thus, a representative sample of sequences can be created simply by generating a large number (but substantially smaller than the total number available) of sequences. From this sample, the most suitable can be selected using the Sohn et al. (1997) criteria. Randomness of the trial sequence is another potential criterion by which to select sequences. This could be done in a similar *sample-then-select* method by selecting sequences on the basis of their conditional entropy, a measure that has been used to assess sequence randomness (e.g. Liu, 2004; Liu & Frank, 2004). Although the above criteria are relevant for some experiments, different criteria may be relevant for other types of experiments. It is impossible to consider all of these here. Other criteria will need to be determined by the experimenter based on the specific details of their experimental design. Regardless of the details, though, it should be straightforward to generate a representative sample of sequences (using the programs provided here) and then select from that sample using the desired criteria.

There are a few important characteristics of and constraints on trial orders generated with the method proposed here. First, each condition is represented equally often. Thus, experiments that involve unequal weighting of conditions cannot be counterbalanced in this way. It may be possible to construct graphs that represent such designs by adjusting the number of vertices and edges. However, any changes may render the graph non-Eulerian and thus no path through it will traverse each edge exactly once. Care will need to be taken to ensure that the graph has edges corresponding to the desired carryover balance and that a path can be found through the graph that results in the desired properties. In order to maintain serial order carryover counterbalancing, additional trials can only be added in multiples of c^2 such that the total number of trials is ac^2+1 where a is the number of repetitions of each ordered pair of conditions and c is the number of conditions. In terms of graph structure, additional trials can be represented by inserting additional copies of every directed edge in the graph. For instance, to double the number of trials, insert exactly one more copy of each edge. The resulting graph will still be Eulerian as long as the in/out degree of each vertex remains equal. Finally, Euler circuits always start and end with the same condition and that condition is repeated once more than all of the other conditions. This extra trial in one condition should likely be excluded from analysis because it represents the only trial that is not counterbalanced for first-order carryover (i.e. it had no trial at all before it). One may worry that this extra trial could amount to extra practice with its condition. With large numbers of repeated measures, however, this effect should be minimal. If this issue is a potential source of confound, then the condition with the extra trial could be counterbalanced over participants.

nth-Order Carryover Effects

The algorithm described above counterbalances for carryover effects from only the immediately preceding condition or trial (first-order carryover effects). However, carryover effects may also arise from conditions presented further back in time. For instance, performance on a trial *t* may differ depending on which conditions were presented on trials *t-1* and *t-2*, i.e. a 2nd-order effect. More generally, any effect of the preceding sequence of *n* trials is an *n*th-order effect. One example of higher order serial carryover effects occurs in *absolute identification* experiments (see recent reviews and discussion in Brown, Marley, & Lacouture, 2007; Matthews & Stewart, 2009; Stewart, 2007; Stewart, Brown, & Chater, 2005). In fact, Stewart et al. (2005) claimed that they know of no absolute identification experiment in which strong sequence effects did not appear. In these experiments a participant is typically presented with a sequence of items that vary along some dimension. For instance, various tones may be presented that vary in loudness with twenty different levels (e.g. Holland & Lockhead, 1968). The participant's task on each trial in this case would be to label the loudness of the tone with a value of 1 to 20. The participant's response on the present trial tends to be *assimilated* to the loudness of the stimulus on the immediately preceding trial (Garner, 1953; Holland & Lockhead, 1968; Lacouture, 1997; Luce, Nosofsky, Green, & Smith, 1982; Staddon, King, & Lockhead, 1980; Ward & Lockhead, 1970, 1971). That is, the value reported on trial *t* tends to be closer to the loudness of the stimulus on trial *t*-1 than it actually was. In contrast, stimuli on earlier trials (e.g. t-2, t-3, t-4, etc.) exert an opposite *contrast* effect on judgments of the current stimulus on trial *t*. That is, participants rate the current stimulus as more different than these trials (Holland & Lockhead, 1968; Lacouture, 1997; Ward & Lockhead, 1970, 1971). Some have found contrast effects from as many as five preceding trials (e.g. Holland & Lockhead, 1968) although there has been disagreement about exactly how far back these effects extend (e.g. Jesteadt, Luce, & Green, 1977). Such effects are well known within this domain and of considerable theoretical interest. Such higher-order carryover effects are often not considered in other domains, though. Therefore it is not clear the extent to which they occur in other areas of behavior. Should they exist, sequences which counterbalance these potential higher-order effects could be useful in evaluating them independently of lower-order and direct effects. Without counterbalancing, one concern is that such higher-order effects could bias condition averages. For instance, in the absolute identification scenario described above, if stimuli of loudness level 10 tend to be preceded more often by sequences of softer stimuli on trials t-2, t-3, and t-4 than by louder stimuli on those trials, then the contrastive effect may bias the participant's estimate of loudness to be higher than it should be. If counterbalancing was done, then louder and softer trials would be equally likely in the range of contrastive effects and thus would be equally reflected in the condition mean. This is the main point of counterbalancing and provides an example of how known higher-order effects could bias measurements.

Higher-order carryover effects can be counterbalanced to the n^{th} order by generating a sequence in which every condition is preceded equally often by every n-tuple (i.e. ordered set of n elements) of conditions. This can be done using a generalization of the first-order solution that has been described above. In n^{th}-order counterbalancing, each vertex in the graph must represent an n-tuple of conditions, i.e. an ordered list of length n containing some combination of the c conditions. Each possible n-tuple is represented by one vertex. Thus, for n^{th}-order counterbalancing of an experiment with c conditions there will be c^n vertices in the graph because there are c^n possible *n*-tuples. This is demonstrated, by example, in Table 6 for 2nd-order counterbalancing. Figure 3 shows an example second-order graph for two conditions.

Unlike in the first-order solution, second-order graphs are not fully-connected. Rather, only vertices which share part of their sequences are linked. Specifically, a vertex sends one directed edge to every other vertex with an *n*-tuple that starts with the same *(n-1)*-tuple with which its own *n*-tuple ends. This includes a loop on vertices which have an *n*-tuple of all the same condition, e.g. “AA”. For instance, for the second-order counterbalancing graph with two conditions in Figure 3, every vertex ending in “A” sends a directed edge to a vertex with an *n*-tuple that starts with “A”. This is a necessary constraint on the connectivity of the graph because it ensures that one can only traverse edges that are consistent with previously traversed conditions. The resulting graph will have equal in and out degree at each vertex. One can see this by examining the example in Table 6 which represents the vertices from a graph with three conditions. Every vertex ending in A, of which there are three, will send an edge to every vertex starting with A, of which there are also three. Every vertex thus has three outbound edges. Every vertex beginning in A, of which there are three, will receive an edge from every vertex ending in A, of which there are also three. Thus, every vertex has three inbound edges. This means that for every outgoing edge from a vertex, there will be an edge returning, i.e. the vertices will have equal in and out degree and therefore the graph is Eulerian. Once this Eulerian graph is constructed, a counterbalanced sequence of trials can be constructed by finding an Euler path through the directed graph, analogously to 1st-order counterbalancing. The first vertex that is visited in the graph contributes all of its conditions to the condition order. Each vertex visited subsequently contributes only its last condition. For instance, if the trail through the second-order graph in Figure 3 visits, in order, vertices AB, BA, AA, AA, AB…, then the resulting condition order will be ABAAAB....

Although this generalization of the counterbalancing solution can address arbitrarily high order carryover effects, there are practical limitations to this. Higher-order counterbalancing involves larger graphs which results in longer trial sequences. Each sequence will be $n + c^{n+1}$ trials long where c is the number of conditions and n is the order of counterbalancing, e.g. $n = 2$ for 2nd-order counterbalancing. There are also limitations on increasing the total number of trials. If n^{th}-order counterbalancing is to be maintained, the overall number of trials can be increased only by inserting into the first sequence additional complete Euler paths through the graph. Thus, the total number of trials must be $n + ac^{n+1}$ where a is the number of repetitions of each n-tuple of conditions. Analogously to first-order counterbalancing, conditions associated with the first vertex that is visited in the graph are over-represented. However, when the first n trials are excluded, all conditions will appear equally often in the remaining sequence. Furthermore, the first n trials are not counterbalanced for n^{th}-order effects because they are preceded by fewer than n conditions. Thus, these trials should be ignored when scoring the data just as the first trial was ignored in first-order counterbalancing.

It is also worth pointing out that n^{th}-order counterbalanced condition orders are counterbalanced for all lower-order carryover effects but only when the first n trials are not considered during analysis (although these trials must still be run during the experiment). Simultaneous lower-order counterbalancing occurs because the vertices of the higher-order graph represent all of the possible combinations/n-tuples of the conditions. For instance, Table 6 shows the 2-tuples that correspond to vertices in a 2nd-order counterbalancing graph with 3 conditions. For second-order counterbalancing, each of these 2-tuples will precede condition A once and condition B once. But notice that within each 2-tuple, each condition on trial N-2 precedes each condition on trial N-1 (including itself) equally often. This is equivalent to first-order counterbalancing within the 2-tuples. Furthermore, because each 2-tuple precedes each condition on trial N equally often (as this is required for 2nd-order counterbalancing) and trial N-1 represents each condition equally often, then trial N will be 1st-order counterbalanced for the influence of the preceding trial. Each of these vertices will be visited equally often in the Euler circuit because the graph is fully-connected and thus all vertices have equal degree (both in-degree and out-degree). Thus, the resulting sequence will be balanced for 2nd-order effects as well as 1st-order effects. Each 1st-order combination will be seen twice in the sequence. An example for 4th-order counterbalancing is shown in Tables S1-S3 in the Supplemental Materials. The sequence is balanced for 4th-order effects. Tables S1-S3 show that 3rd, 2nd, and 1st-order effects are also balanced because each condition is preceded by every 3, 2, and 1-tuple equally often. Again though, this depends on removing the first *n* trials from consideration during analysis.

Purposefully Unbalanced Orders

If serial order carryover effects are of particular interest in an experiment, it is also possible to build condition orders that purposefully *include* carryover biases. This can be done within the method presented above by adjusting the relative number of edges between the vertices representing the conditions of interest. For instance, if the experiment involves A preceding B twice as often as C precedes D then there should be twice as many edges between A and B as between C and D. However, the changes to the graph must maintain equal in and out degree of each vertex in order for there to be an Eulerian circuit through the graph. If an Eulerian circuit is not available, it may be possible to find an Eulerian trail (which does not start and stop at the same place but still traverses every edge exactly once). In this case the graph is sometimes called *semi*-Eulerian. An Eulerian trail exists if exactly one vertex has an out-degree one greater than its in-degree, exactly one vertex has in-degree one greater than its out-degree, and all other vertices have equal in and out degree. The trail must start at the vertex with out-degree greater than in-degree. In general, there are many potential graphs that could be constructed to represent trial sequences. For some applications, graphs may not need to be Eulerian. Other experimental requirements may be satisfied by other types of paths through the graph. Graph theory provides a large set of tools for finding these.

Practical Implementation

Practical implementation of this method for generating sequences of trials involves: (1) Construct a directed graph representation of the conditions and their temporal relationships. Graphs are often represented in software by their *adjacency matrix*, i.e. a $c \times c$ (where c is the number of vertices) matrix with a positive integer in cell c_{ij} representing the number of directed edges from vertex i to vertex j. (2) Verify that this graph is Eulerian (or semi-Eulerian). (3) Find an Euler Circuit or trail through the graph. This can be done in any programming language using the algorithms described in Tables 4 or 5 and may be easiest in high-level languages such as Matlab (Mathworks, Inc., Natick, Massachusetts, http://www.mathworks.com), Octave (http://www.gnu.org/software/octave/), R (http:// http://www.r-project.org/), or Mathematica (Wolfram Research, Inc., Mathematica, Version 8.0, Champaign, IL., http://www.mathematica.com). In fact, Mathematica has built in functions for determining whether a graph is Eulerian (function name: *EulerianGraphQ*) and for finding an Euler path through a graph (function name: *FindEulerianCycle*).

The Supplemental Material section contains a Matlab/Octave function which can generate counterbalanced trial sequences for any number of conditions less than 100. It can counterbalance for any order of counterbalancing (e.g. 1st-order, 2nd-order, nth-order) although very high orders of counterbalancing may lead to very long sequences and exceed the memory of the system. It can also handle repetitions which can be used to extend the length of the sequences while maintaining counterbalancing. There is also an option for omitting condition self-adjacencies thus providing an analog for the Balanced Latin-Square and Finney & Outhwaite Type 2 sequences which do not contain self-adjacencies (Finney & Outhwaite, 1955, 1956). Instructions for use are included in the Supplemental Materials. The Supplemental Materials also includes a link to an executable version of this function. Both of these programs implement the Kandel et al. (1996) method for finding Euler paths.

Discussion

All psychology and neuroscience experiments that involve more than one condition seen by each participant have the potential for serial order carryover effects. Partial counterbalancing procedures have long provided a way to reduce potential serial order carryover effects as well as the effects of position in the sequence. Poor carryover counterbalancing on the individual participant level may add noise to experiments and, at worst, may be an unknown confound. Here, I have presented a method for fully counterbalancing trial and condition orders against carryover effects in repeated-measures designs with multiple measurements of each condition. This method relies on representing conditions as vertices in a graph and representing the temporal relationships between them as directed edges. Serially balanced sequences can be found as Euler circuits through the graph in which each edge (i.e. temporal relationship) is traversed once (or equally often when more measurements are desirable). There are existing algorithms for finding Euler circuits. Importantly, this method can also be generalized to address higher-order carryover effects. There is also potential to create condition orders with other properties, such as intentionally unbalanced orders. Reframing the condition ordering problem in terms of graph theory makes it possible to use the known properties of graphs to generate sequences with any property for which there exists a graph theory solution. Beyond this, the method presented in this paper suggests a general framework for imposing constraints of any type on condition orders. Once conditions are represented as vertices in a graph, edges can represent any relationship between those conditions. By varying the connectivity of the graph representation and the type of path taken through it, researchers can select condition sequences with a wide array of properties. Graph theory provides a toolkit for exploring the possibilities.

Other methods of serial balancing such as balanced Latin squares, m-sequences, and the sequences of Finney and Outhwaite (1955) also exist. In some cases, the serially balanced sequences generated as proposed here may be preferable. Latin squares are better suited for relatively short sequences with no repetitions of a condition within each sequence. With longer sequences, many participants or blocks will be needed. The method proposed in this paper can be used to generate sequences containing multiple measurements of the same condition within the same participant or block. Finney and Outhwaite sequences have a block pattern to them in which the positions of condition self-adjacencies are fixed. The sequences proposed here do not have this constraint and are not predictable (although they do start and end with the same condition). Both m-sequences and Finney and Outhwaite sequences are unavailable for a few numbers of conditions. The sequences proposed here are available for any number of conditions. m-sequences are also only approximately serially balanced whereas the sequences proposed here are exactly balanced. Finally, the graph model proposed here can be extended to include higher-order serial balancing.

Of course, there are potential drawbacks of using the sequences proposed here. Repeated measurements can only be added in certain multiples which may lead to longer or shorter experiments than desired. Each sequence contains an extra trial of one condition (for first-order sequences) which may lead to extra practice in that condition. Unlike the Finney and Outhwaite sequences, the sequences proposed here do not appear in blocks which also balance the approximate position of conditions across the whole sequence. Balancing of approximate position in the sequence may be particularly desirable in memory experiments, for instance, in which serial position is known to have a dramatic effect on later memory of items (e.g., Deese & Kaufman, 1957). Some of these drawbacks may be overcome, however. For instance, the extra trial for one condition can be neutralized by counterbalancing which condition is over-represented across subjects. Approximate position may be counterbalanced by using different sequences across participants or by selecting the most balanced sequences from amongst a random sample of sequences that was produced. The sequences proposed in this paper are not meant to solve all problems simultaneously but they do present sometimes preferable alternative to previous proposals for generating serially balanced sequences.

Serially-balanced sequences like those created by the method here (as well as some of the alternatives mentioned above) ensure that every condition is preceded by every other condition (including itself) equally often. For *n*th-order counterbalancing, every condition is preceded by every *n*-tuple of conditions equally often. This can be achieved from a single run through the sequence in one participant. Recall from comparing Tables 1 and 2, that if serial-order carryover effects are present and they are unbalanced then the resulting condition averages (when collapsed over which trial(s) preceded it) can be biased if the condition is preceded by some conditions more often than others (e.g. Table 2). Counterbalancing does not remove any serial carryover effects that are present, it just ensures that all of the possible different carryover effects (arising from any of the preceding *n*-tuples of conditions) are equally represented when contributing to the average for each condition. If the carryover effects are of further interest, then given a counterbalanced design, a factor representing the previous trial/*n*-tuples could be added to the analysis and these effects could be examined independently of the direct effects of each condition.

In Tables 1 and 2, the carryover effects were represented as constant across the course of the experiment. That is, although the effect in condition A differed depending on the preceding trial, this carryover effect was consistent across time. This is probably not realistic as in many experiments practice effects will occur over the course of the experiment. Nonetheless, the sequence will still be balanced such that each condition average will comprise instances preceded equally often by each condition. However, for sequences without repetitions of the *n*-tuples, some of the *n*-tuples (a condition preceded by one or set of other conditions) will only occur at the beginning of the sequence whereas others will only occur in the later half. If a carryover effect changes over time and the *n*-tuples that detect that carryover effect are not evenly distributed across the sequence, then the carryover effect could be mischaracterized even with a serially-counterbalanced sequence. This effect could be ameliorated by using different sequences across different participants and then averaging across them making it more likely that each n-tuple is represented equally often in each serial position in the sequence. Alternatively, one may use sequences containing repetitions of each n-tuple and then selecting those in which the instances of the n-tuples are best distributed. This later solution would ensure that each possible carryover effect is assessed equally-often across the sequence. This could be done within the sequence for a single participant. The method for assessing sequence serial balance proposed by Sohn et al. (1997) could be used here.

Another possible problem is that the presentation of one condition may change a participant's strategy and thus have long-lasting effects across the experiment. For instance, imagine a visual perception experiment with ten conditions and a within-subjects design. In nine of the conditions, participants see an array of blocks with letters depicted in the space between the blocks as shown in Figure 4A (stimuli, but not the task, derived from Davis, Schiffman, & Greist-Bousquet, 1990). Sometimes these hidden letters depict words and sometimes they depict non-words. Participants then name a subsequent picture of an object which is either related or unrelated to the previously seen word or letter string. These conditions could test for subliminal semantic priming from the hidden words. The tenth condition contains clearly discernable block letters in the prime (Figure 4B) and this is meant to test for non-subliminal effects. Inclusion of this tenth condition, however, may alert participants to the potential presence of letters in the other conditions. Even after just one presentation of this tenth condition, the participant's strategy may change. They may begin looking for letters and words in the stimuli. This is likely to affect performance on all subsequent instances of the other conditions. This type of carryover effect or *asymmetric transfer effect* (Poulton, 1982) has potential to affect the internal and external validity of the experiment. That is, some conditions may function differently in the context of one set of conditions than when presented alone or in the context of different conditions. Counterbalancing, whether with the sequences proposed here or with other methods, will not be able to remove this effect. It may be possible to model the effects as very high-order carryover effects but this is unlikely to be practical in long sequences. Depending on the goals of the experiment, it may be better to use a between-subjects or mixed design. The use of counterbalancing, of any type, does not absolve the experimenter from carefully considering their experimental design and how any likely carryover effects may impact the interpretation of their results. The sequences suggested here do allow one, however, to be sure that every condition mean is computed from equal numbers of trials preceded by every n-tuple of conditions (for an nth-order counterbalanced sequence) equally often. Any inference about a different type of carryover effect will likely require a different experimental design aimed specifically at addressing that issue.

When giving examples, I have mostly discussed applications within cognitive psychology or cognitive neuroscience. However, the sequences generated here can be used in domains in which a within-subjects design with multiple repetitions of each condition will be presented. It may not be suitable for some domains though. For instance, in questionnaire design, each item may appear only once in the questionnaire. In this case, it is not possible to counterbalance carryover effects from one question to the next within a single questionnaire. This occurs because if question A appears only once then it can only precede one other condition and not the others which would be necessary for serial carryover counterbalance. In this case, counterbalancing will need to be achieved between-subjects. However, if a questionnaire involves several different items that assess the same underlying target/condition (e.g. self esteem) then it may be possible to use the current method to generate sequences of questions that are serially counterbalanced. It is important to note though, that in any such sequence, any differences due to different questions will be confounded with the carryover effects. That is, although the different questions are meant to assess the same underlying construct, they may do this to different extents or measure slightly different aspects of it. This could be counterbalanced between-subjects by using different sequences in different participants. More generally, it will be important to determine which effects are of interest and then to ensure that they are evaluated independently of any carryover effects that are counterbalanced.

The serially balanced sequences generated from the graph representations proposed in this paper cannot eliminate serial order carryover effects. However, they can reduce the influence of these effects on comparisons of interest or allow the carryover effects to be assessed orthogonally to the other manipulations. In order to completely remove serial order carryover effects, other changes will need to be made to the experimental design. For instance, *washout periods* can be introduced between experimental conditions to remove lingering effects from the preceding condition. Where this is not possible, counterbalancing can be used. The serially balanced sequences described here provide a tool for achieving this goal and open up a general approach based on graph representations for generating sequences with useful properties for psychological and neuroscience research.

References

Aardenne-Ehrenfest, T. van, & de Bruijn, N. G. (1951). Circuits and trees in oriented linear graphs. *Simon Stevin*, *28*, 203-217.

Aguirre, G. K. (2007). Continuous carry-over designs for fMRI. *NeuroImage*, *35*(4), 1480-1494. doi:10.1016/j.neuroimage.2007.02.005

Bartram, D. (1974). The role of visual and semantic codes in object naming. *Cognitive Psychology*, *6*(3), 325-356. doi:10.1016/0010-0285(74)90016-4

Baseler, H. A., Sutter, E. E., Klein, S. A., & Carney, T. (1994). The topography of visual evoked response properties across the visual field. *Electroencephalography and clinical neurophysiology*, *90*(1), 65-81.

Baylis, G. C., & Cale, E. M. (2001). The figure has a shape, but the ground does not: evidence from a priming paradigm. *Journal of experimental psychology. Human perception and performance*, *27*(3), 633-643.

Bernadette, E. A., & Victor, J. D. (1994). An extension of the m-sequence technique for the analysis of multi-input nonlinear systems. In V. Z. Marmarelis (Ed.), *Advanced methods of physiological system modeling, Volume 3* (pp. 87-110). New York: Plenum Press.

Biederman, I., & Cooper, E. E. (1991a). Priming contour-deleted images: evidence for intermediate representations in visual object recognition. *Cognitive psychology*, *23*(3), 393-419.

Biederman, I., & Cooper, E. E. (1991b). Evidence for complete translational and reflectional invariance in visual object priming. *Perception*, *20*(5), 585-593.

Biggs, N., Lloyd, E., & Wilson, R. (1986). *Graph Theory, 1736-1936*. *Graph Theory, 1736-1936*. New York: Oxford University Press.

Blakemore, C., & Campbell, F. W. (1969). Adaptation to spatial stimuli. *The Journal of physiology*, *200*(1), 11P-13P.

Blakemore, C., & Nachmias, J. (1971). The orientation specificity of two visual after-effects. *The Journal of physiology*, *213*(1), 157-174.

Bradley, A., Switkes, E., & De Valois, K. (1988). Orientation and spatial frequency selectivity of adaptation to color and luminance gratings. *Vision research*, *28*(7), 841-856.

Bradley, J. (1958). Complete Counterbalancing of Immediate Sequential Effects in a Latin Square Design. *Journal of the American Statistical Association*, *53*(282), 525- 528.

Brimijoin, W. O., & O'Neill, W. E. (2010). Patterned tone sequences reveal non-linear interactions in auditory spectrotemporal receptive fields in the inferior colliculus. *Hearing research*, *267*(1-2), 96-110. doi:10.1016/j.heares.2010.04.005

Brown, S., Marley, A. A., & Lacouture, Y. (2007). Is absolute identification always relative? Comment on Stewart, Brown, and Chater (2005). *Psychological Review, 114*(2), 528-32. doi:10.1037/0033-295X.114.2.528

Buracas, G. T., & Boynton, G. M. (2002). Efficient design of event-related fMRI experiments using M-sequences. *NeuroImage*, *16*(3), 801-813.

Creed, P., & Cryan, M. (2010). The number of Euler tours of a random d-in/d-out graph. *DMTCS Proceedings* (pp. 117-128).

Davis, J., Schiffman, H. R., & Greist-Bousquet, S. (1990). Semantic context and figure-ground organization. *Psychological research*, *52*(4), 306-9.

Deese, J., & Kaufman, R. A. (1957). Serial effects in recall of unorganized and sequentially organized verbal material. *Journal of experimental psychology*, *54*(3), 180-187.

Duriera, C., Monoda, H., & Bruetschy, A. (1997). Design and analysis of factorial sensory experiments with carry-over effects. *Food Quality and Preference, 8*(2), 141-149.

Edwards, A. L. (1951). Balanced latin-square designs in psychological research. *Psychological Research*, *64*(4), 598-603.

Elmes, D. G., Kantowitz, B. H., & Roediger, H. L. (1999). *Research Methods in Psychology* (6th ed.). Pacific Grove, California: Brooks/Cole Publishing Company.

Esterman, M., McGlinchey-Berroth, R., Verfaellie, M., Grande, L., Kilduff, P., & Milberg, W. (2002). Aware and unaware perception in hemispatial neglect: evidence from a stem completion priming task. *Cortex*, *38*(2), 233-246.

Euler, L. (1736). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, *8*, 128-140.

Fang, F., Boyaci, H., & Kersten, D. (2009). Border ownership selectivity in human early visual cortex and its modulation by attention. *The Journal of Neuroscience, 29*(2), 460-465. doi:10.1523/JNEUROSCI.4628-08.2009

Faulkner, K. K., & Cogan, R. (1990). Measures of shame and conflict tactics: effects of questionnaire order. *Psychological Reports*, *66*(3, Pt 2), 1217-8.

Finney, D. J., & Outhwaite, A. D. (1955). Serially Balanced Sequences. *Nature*, *176*, 748.

Finney, D. J., & Outhwaite, A. D. (1956). Serially Balanced Sequences in Bioassay. *Proceedings of the Royal Society B: Biological Sciences*, *145*(921), 493-507. doi:10.1098/rspb.1956.0058

Fize, D., Boulanouar, K., Chatel, Y., Ranjeva, J. P., Fabre-Thorpe, M., & Thorpe, S. (2000). Brain areas involved in rapid categorization of natural images: an event-related fMRI study. *NeuroImage*, *11*(6), 634-643. doi:10.1006/nimg.2000.0585

Fleischner, H. (1991). Algorithms for Eulerian Trails. *Eulerian Graphs and Related Topics: Part 1, Volume 2* (pp. X.1–13). Amsterdam: Elsevier.

Fleury, M. (1883). Deux problemes de geometrie de situation. *Journal de mathematiques elementaires*, 257-261.

Fortune, B., Demirel, S., & Bui, B. V. (2009). Multifocal visual evoked potential responses to pattern-reversal, pattern-onset, pattern-offset, and sparse pulse stimuli. *Visual neuroscience*, *26*(2), 227-235. doi:10.1017/S0952523808080954

Gama, H., Correia, S., & Lunet, N. (2009). Effect of questionnaire structure on recall of drug utilization in a population of university students. *BMC medical research methodology*, *9*(1), 45. doi:10.1186/1471-2288-9-45

Garner, W. R. (1953). An informational analysis of absolute judgments of loudness. *Journal of Experimental Psychology*, *46*(5), 373-80.

Gilaie-Dotan, S., Gelbard-Sagiv, H., & Malach, R. (2010). Perceptual shape sensitivity to upright and inverted faces is reflected in neuronal adaptation. *NeuroImage*, *50*(2), 383-395. doi:10.1016/j.neuroimage.2009.12.077

Goldstein, A. G. (1958). On the after-effects of the waterfall and spiral illusions. *The American journal of psychology*, *71*(3), 608-609.

Golomb, S. (1967). *Shift register sequences*. San Francisco: Holden-Day.

Grill-Spector, K., & Malach, R. (2001). fMR-adaptation: a tool for studying the functional properties of human cortical neurons. *Acta psychologica*, *107*(1-3), 293-321.

Haushofer, J., Baker, C. I., Livingstone, M. S., & Kanwisher, N. (2008). Privileged Coding of Convex Shapes in Human Object-Selective Cortex. *Journal of Neurophysiology*, *100*(2), 753-762. doi:10.1152/jn.90310.2008

Henson, R. N., & Rugg, M. D. (2003). Neural response suppression, haemodynamic repetition effects, and behavioural priming. *Neuropsychologia*, *41*(3), 263-270.

Hierholzer, C. (1873). Ueber die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren. *Mathematische Annalen*, *6*(1), 30-32.

Holland, M. K., & Lockhead, G. R. (1968). Sequential effects in absolute judgments of loudness. *Perception & Psychophysics*, *3*(6), 409-414. doi:10.3758/BF03205747

Jesteadt, W., Luce, R. D., & Green, D. M. (1977). Sequential effects in judgments of loudness. *Journal of experimental psychology. Human perception and performance*, *3*(1), 92-104.

Kandel, D., Matias, Y., Unger, R., & Winkler, P. (1996). Shuffling biological sequences. *Discrete Applied Mathematics*, *71*(1-3), 171-185.

Krekelberg, B., Boynton, G. M., & van Wezel, R. J. A. (2006). Adaptation: from single cells to BOLD signals. *Trends in neurosciences*, *29*(5), 250-6. doi:10.1016/j.tins.2006.02.008

Lacouture, Y. (1997). Bow, range, and sequential effects in absolute identification: a response-time analysis. *Psychological Research*, *60*(3), 121-33.

Li, Y., Bin, G., Hong, B., & Gao, X. (2010). A coded VEP method to measure interhemispheric transfer time (IHTT). *Neuroscience letters*, *472*(2), 123-127. doi:10.1016/j.neulet.2010.01.069

Liu, T. T. (2004). Efficiency, power, and entropy in event-related fMRI with multiple trial types. Part II: design of experiments. *NeuroImage*, *21*(1), 401-13.

Liu, T. T., & Frank, L. R. (2004). Efficiency, power, and entropy in event-related FMRI with multiple trial types. Part I: theory. *NeuroImage*, *21*(1), 387-400.

Luce, R. D., Nosofsky, R. M., Green, D. M., & Smith, A. F. (1982). The bow and sequential effects in absolute identification. *Perception & Psychophysics*, *32*(5), 397-408.

Làdavas, E., Paladini, R., & Cubelli, R. (1993). Implicit associative priming in a patient with left visual neglect. *Neuropsychologia*, *31*(12), 1307-1320.

Matthews, W. J., & Stewart, N. (2009). The effect of interstimulus interval on sequential effects in absolute identification. *Quarterly journal of experimental psychology*, *62*(10), 2014-29. doi:10.1080/17470210802649285

McHugh, J. A. M. (1989). *Algorithmic graph theory*. Upper Saddle River, NJ: Prentice-Hall.

Miller, J. M., & Krosnick, J. A. (1998). The Impact of Candidate Name Order on Election Outcomes. *Public Opinion Quarterly*, *62*(3), 291. doi:10.1086/297848

Muir, D. D., & Hunter, E. A. (1992). Sensory evaluation of Cheddar cheese: Order of tasting and carryover effects. *Food Quality and Preference*, *3*(3), 141-145.

Nonyane, B. a S., & Theobald, C. M. (2007). Design sequences for sensory studies: achieving balance for carry-over and position effects. *The British journal of mathematical and statistical psychology*, *60*, 339-349. doi:10.1348/000711006X114568

Pashler, H., & Baylis, G. (1991a). Procedural Learning: 1. Locus of Practice Effects in Speeded Choice Tasks. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *17*(1), 20-32.

Pashler, H., & Baylis, G. (1991b). Procedural Learning: 2. Intertrial Repetition Effects in Speeded-Choice Tasks. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *17*(1), 33-48.

Pasley, B. N., Allen, E. A., & Freeman, R. D. (2009). State-dependent variability of neuronal responses to transcranial magnetic stimulation of the visual cortex. *Neuron*, *62*(2), 291-303. doi:10.1016/j.neuron.2009.03.012

Paterson, L. (1983). Circuits and efficiency in incomplete block designs. *Biometrika*, *70*(1), 215-225. doi:10.1093/biomet/70.1.215

Poulton, E. C. (1982). Influential companions: Effects of one strategy on another in the within-subjects designs of cognitive psychology. *Psychological Bulletin*, *91*(3), 673-690.

Sampford, M. R. (1957). Methods of Construction and Analysis of Serially Balanced Sequences. *Journal of the Royal Statistical Society, B*, *19*(2), 286-304.

Schlich, P. (1993). Uses of change-over designs and repeated measurements in sensory and consumer studies. *Food Quality and Preference*, *4*(4), 223-235.

Silvanto, J., Muggleton, N., & Walsh, V. (2008). State-dependency in brain stimulation studies of perception and cognition. *Trends in cognitive sciences*, *12*(12), 447-454. doi:10.1016/j.tics.2008.09.004

Sohn, H.-S., Bricker, D. L., Simon, J. R., & Hsieh, Y.-chih. (1997). Optimal sequences of trials for balancing practice and repetition effects. *Behavior Research Methods, Instruments, & Computers*, *29*(4), 574-581. doi:10.3758/BF03210610

Staddon, J. E., King, M., & Lockhead, G. R. (1980). On sequential effects in absolute judgment experiments. *Journal of Experimental Psychology: Human Perception and Performance*, *6*(2), 290-301.

Stewart, N. (2007). Absolute identification is relative: a reply to Brown, Marley, and Lacouture (2007). *Psychological Review*, *114*(2), 533-8. doi:10.1037/0033-295X.114.2.533

Stewart, N., Brown, G. D. A., & Chater, N. (2005). Absolute identification by relative judgment. *Psychological Review*, *112*(4), 881-911. doi:10.1037/0033-295X.112.4.881

Street, A. P., & Street, D. J. (1987). *Combinatorics of Experimental Design*. Oxford: Clarendon.

Tutte, W. T., & Smith, C. A. B. (1941). On Unicursal Paths in a Network of Degree 4. *American Mathematical Monthly*, *48*, 233-237.

Vuilleumier, P., Henson, R. N., Driver, J., & Dolan, R. J. (2002). Multiple levels of visual object constancy revealed by event-related fMRI of repetition priming. *Nature neuroscience*, *5*(5), 491-499. doi:10.1038/nn839

Wakeling, I. N., & MacFie, H. J. H. (1995). Designing consumer trials balanced for first and higher orders of carry-over effect when only a subset of k samples from t may be tested. *Food Quality and Preference*, *6*(4), 299-308.

Ward, L. M., & Lockhead, G. R. (1970). Sequential effects and memory in category judgments. *Journal of Experimental Psychology*, *84*(1), 27-34.

Ward, L. M., & Lockhead, G. R. (1971). Response system processes in absolute judgment. *Perception & Psychophysics*, *9*(1), 73-78. doi:10.3758/BF03213031

West, D. B. (2001). *Introduction to graph theory* (2nd ed.). Upper Saddle River, NJ: Prentice-Hall, Inc.

Williams, E. J. (1949). Experimental designs balanced for the estimation of residual effects of treatments. *Australian Journal of Scientific Research, A, 2*, 149-168.

Williams, R. M. (1952). Experimental Designs for Serially Correlated Observations. *Biometrika*, *39*(1), 151-167.

Appendix

Counting the number of all possible unique condition sequences

This Appendix presents a method for calculating the number of all possible unique condition sequences that can be generated using the graph method described in the main text of this paper. Using this method, an Eulerian directed graph serves to represent experimental conditions and the temporal relationships between them. Because an Eulerian path through the graph represents a counterbalanced condition sequence, counting the number of possible condition sequences is related to the number of unique Eulerian paths through the graph. This number can then be used to calculate the number of possible unique condition sequences as described below.

For an Eulerian directed graph, *G* = (*V*,*E*), where *V* is the set of all vertices (representing conditions in an experiment) in *G* and *E* is the set of all directed edges (representing temporal relationships between conditions) in *G*, the total number of Eulerian circuits through *G*, denoted ec(*G)*, can be calculated according to the BEST theorem (Aardenne-Ehrenfest & de Bruijn, 1951; Creed & Cryan, 2010; Tutte & Smith, 1941) as:

$$\textit{Equation A1:}\quad ec\ \ G\ \ = t_w G \prod_{v \in V} (\deg_{out}(v) - 1)!$$

The first term in Equation A1, $t_w G$, is the number of *arborescences* in *G* rooted at any vertex *w*. An arborescence is a directed graph in which, from a given root vertex, there is exactly one directed path to each other vertex, *v*. The number of arborescences can be calculated as a *determinant*, det(**L***), where **L** is the Laplacian matrix (also called admittance or Kirchhoff matrix) of *G*. The Laplacian matrix, **L**, is calculated as **L** = **D**-**A** where **D** and **A** are, respectively, the degree matrix and the adjacency matrix of the graph, *G*. The matrix **L*** is a minor of the Laplacian matrix, **L**, formed by removing any row and any column from **L**. The *degree matrix* is a $c \times c$ (c is the number of conditions/vertices) matrix that represents, as positive integers along the diagonal of the matrix, the out-degree, $\deg_{out}$, (number of outgoing edges) for each vertex, v, in V. The *adjacency matrix* is a $c \times c$ matrix that represents the connectivity of the graph using a positive integer at position $c(i,j)$ to represent the number of directed edges going from vertex v_i to vertex v_j. Below is a worked example of this calculation for the graph in Figure 1B.

Calculation 1:

$$\mathbf{L} = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix} = \mathbf{D} - \mathbf{A} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\mathbf{L}^* = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix}$$

$$t_w G = \det \ \mathbf{L}^* \ = 16$$

The determinant can be easily calculated using software such as Matlab (Mathworks; Natick, MA, USA; http://www.mathworks.com/), Octave (http://www.gnu.org/software/octave/), or Mathematica (Wolfram Research; Champaign, IL, USA; http://www.wolfram.com/mathematica/). Any Linear Algebra textbook will have a description of how to compute a determinant.

The second term in Equation A1 is the product across all vertices, of the out-degree, $\deg_{out}$, of each vertex minus 1, factorial. For the graph in Figure 1B, all four vertices have an out-degree of 4 and therefore the calculation is:

Calculation 2:

$$\prod_{v \in V} \deg_{out} \ v \ -1 \ != \ 4-1 \ ! \ 4-1 \ ! \ 4-1 \ ! \ 4-1 \ ! = 1296$$

Therefore, taking the product of the results from Calculations 1 & 2 above, the total number of Eulerian circuits through the graph, ec(G), in Figure 1B is 16 × 1296 = 20736. This quantity, *ec*(G), however, does not represent the total number of trial sequences, *seq*(G), that can be generated from graph G. Each Euler circuit counted in *ec*(G) is an ordered, cyclically continuous set of transitions around the edges of G. For example, consider the Eulerian graph in Figure A1-A with 3 vertices and 6 edges. One possible Eulerian circuit through this graph is represented by the continuous cycle in Figures A1-B and A1-C. Notice that different trial sequences can be derived from the same Eulerian circuit depending on which vertex is used as the starting/ending point. Thus, because Eulerian circuits are cyclic and the start/end point is not considered when enumerating them, *ec*(G) undercounts the number of possible sequences, seq(G). Equation A2 multiplies Equation A1 by a third term to account for these additional variations on the Eulerian circuits. This term is equivalent to the number of edge traversals in the circuit counted as the sum of the out-degrees of all of the vertices.

$$\textit{Equation A2:}\quad seq(G) = t_w G\left(\prod_{v\in V} \deg_{out}\ v\ -1\ !\right)\sum_{v\in V}\deg_{out}\ v$$

Counting the number of unique sequences in graphs with multiplicities of edges

Equation A2 applies in situations where each edge, e_{vw}, in E (the set of edges of graph G) occurs at most once. In some cases one may want to introduce multiple instances of edges in order to increase the total number of trials. For instance, duplicating each edge will double the number of trials while maintaining counterbalancing. Duplicate edges are psychologically equivalent to one another. Traversing an edge, $e1_{1,2}$, from vertex v_1 to v_2 is psychologically equivalent to traversing its duplicate, $e2_{1,2}$. However, Equation A2 counts these as different for purposes of counting the number of Euler circuits through the graph. For instance, consider the subsequence BBB. This could correspond to the ordered set of edges {*e1*$_{B,B}$, *e2*$_{B,B}$} or {*e2*$_{B,B}$, *e1*$_{B,B}$} and these would be counted as different paths by Equation A2 despite being psychologically equivalent. Specifically, equivalent sequences are produced by permuting the repeated out-edges of a particular vertex. This is corrected in Equation A3 with an additional term. For each edge, *e* in *E*, $reps_e$ is the number of instances of that edge in *E*.

$$\textit{Equation A3:}\;\; seq(G) = t_w G\left(\prod_{v \in V}(\deg_{out}(v)-1)!\right)\sum_{v \in V}\deg_{out}(v)\left(\frac{1}{\prod_{e \in E} reps_e\,!}\right)$$

Author Notes

Joseph L. Brooks, UCL Institute of Cognitive Neuroscience, University College London. J.L.B. was supported by post-doctoral fellowships from the Royal Society and the British Academy. I thank David Oppenheimer, Ph.D. for fruitful discussions, Páidí Creed, Ph.D. at the School of Mathematical Sciences at Queen Mary, University of London for his generous help with the details in the Appendix, and Alexandra List, Ph.D. who presented this problem to me for a solution

Correspondence concerning this article should be addressed to Joseph L. Brooks, UCL Institute of Cognitive Neuroscience, University College London, 17 Queen Square, London, WC1N 3AR, United Kingdom. E-mail: joseph.brooks@ucl.ac.uk

Table 1

Serial order carryover effects counterbalanced in trial order

Trial Number	Condition	Trial N = Condition A		Trial N = Condition B	
		Trial N-1 A	Trial N-1 B	Trial N-1 A	Trial N-1 B
1	A	-[a]	-[a]	-[a]	-[a]
2	B			50	
3	B				50
4	A		10		
5	A	100			
6	A	100			
7	B			50	
8	B				50
9	A		10		
10	A	100			
11	B			50	
12	B				50
13	A		10		
14	B			50	
15	B				50
16	A		10		
17	B			50	
18	B				50
19	A		10		
20	A	100			
21	A	100			
	Average	Condition A 55		Condition B 50	

Note: The score for condition A depends on the preceding condition: condition A preceded by A (Trial N = A and Trial N-1 = A, 4th column from right) has a score of 100 whereas A preceded by B (Trial N = A and Trial N-1 = B, 3rd column from right) has a score of 10. The condition B score is independent of the preceding condition (always 50). Scores are assigned in the 4 right columns on the basis of the condition and the one that preceded it. At the bottom of the table the scores are given for condition A and condition B collapsed over the serial order. Because first-order serial order effects are counterbalanced in this trial sequence the mean for condition A is 55. This reflects the average score in condition A collapsed over any effects of serial order carryover from the preceding condition. On average condition A is greater than condition B but this cannot be attributed to an unbalanced carryover effect.

[a] The first trial can be omitted from the calculation of the means because it is not preceded by any other stimulus. It is present to serve as preceding stimulus for the following condition. Including it would unbalance the number of measurements in conditions A and B

Table 2

Unbalanced serial order carryover effects

Trial Number	Condition	Trial N = Condition A		Trial N = Condition B	
		Trial N-1 A	Trial N-1 B	Trial N-1 A	Trial N-1 B
1	A	-[a]	-[a]	-[a]	-[a]
2	A	100			
3	A	100			
4	A	100			
5	A	100			
6	B			50	
7	A		10		
8	B			50	
9	A		10		
10	A	100			
11	A	100			
12	A	100			
13	B			50	
14	B				50
15	B				50
16	B				50
17	B				50
18	A		10		
19	B			50	
20	B				50
21	B				50
	Average	Condition A 73		Condition B 50	

Note: The format and size of serial order effects are the same as in Table 1. Unlike Table 1, first-order serial order effects are unbalanced in this trial sequence. The average score for condition A is inflated due to over-representation of A trials preceded by other A trials (Trial N = A and Trial N-1 = A, 4th column from right). These AA trials have a higher score and thus inflate the mean when collapsed with condition A trials which were preceded by condition B (Trial N = A and

Trial N-1 = B, 3rd column from right). The mean in each condition is computed across all 10 scores.

[a] The first trial was omitted from mean calculations. See Table 1 notes for explanation.

Table 3

A balanced Latin square with four conditions

Participant	Trial >	1	2	3	4
1		A	B	D	C
2		B	C	A	D
3		C	D	B	A
4		D	A	C	B

Table 4

Hierholzer's Algorithm for finding Euler circuits

Step Number	Description
1	Randomly choose a starting vertex *v* in a Eulerian directed graph, *G* = (*V*,*E*). Add *v* to the ordered set of vertices, *T*.
2	Choose randomly an untraversed edge, e_{vw}, exiting *v* to vertex *w*. Traverse the edge. Mark this edge as traversed and do not use again. Add *w* to the ordered set of vertices *T*.
3	Starting at vertex *w*, repeat step 2 until there are no directed edges exiting from the current vertex. This is the end of trail *T*. If *T* contains all the edges of *G* then *T* is an Euler circuit of *G* thus go to Step 9. Otherwise go to Step 4.
4	Randomly chose a new starting vertex, *T*(*v*), visited in previously-created trail, *T*, which also has previously-untraveled edges exiting the vertex.
5	Choose randomly an untraversed edge, e_{vw}, exiting *v* to vertex *w*. Traverse the edge. Mark this edge as traversed and do not use again. Add *w* to the ordered set of vertices *T'*.
6	Repeat step 5 until no exiting edges
7	Replace vertex *T(v)* in the trail *T* with the trail *T'*.
8	Repeat (4-8) until all edges in *G* have been traversed.
9	The final trial *T* is an Euler path through *G*

Note: *V* and *E* are the set of vertices and set of edges of *G*, respectively. The algorithm described here was first described by Hierholzer (1873) and the version here is based on the English descriptions by Fleischner (1991) and McHugh (1989).

Table 5

Kandel et al.'s (1996) *algorithm for finding Euler circuits*

Step Number	Description
1	Randomly choose a starting vertex *v* in a Eulerian directed graph, *G* = (*V*,*E*)
2	Perform a backward random walk starting at *v*. (a) Choose randomly an edge, e_{wv}, entering *v* from some other vertex *w*. If this is the first time that *w* has been visited add this edge to set *A*. (b) Repeat step (a) until every vertex has been visited at least once. (c) The resulting set of edges, *A*, form an arborescence of *G*.
3	At each vertex, *v* in *V*, label each outgoing edge, e_{vw}, with a random number with the exception that any edge amongst these that is in *A* should be given the highest value number.
4	For *v* being any randomly-selected vertex in *V*, start at origin vertex *v* and add *v* to the ordered set of vertices, *T*.
5	Choose the untraversed edge, e_{vw}, from origin vertex, *v*, to target vertex, *w*, with the lowest number (assigned in step 3) amongst the untraversed edges exiting *v*. Traverse this edge. Add the target vertex *w* to *T*. Mark edge e_{vw} as traversed and do not traverse this edge again.
6	Set the origin vertex *v* in step 5 as *w* which was the target vertex in step 5. Repeat Step 5 to choose a new edge e_{vw} with new target *w*. Repeat until all edges in *E* have been traversed.
7	*T* is an Euler path through *G*

Note: *V* and *E* are the set of vertices and set of edges of *G*, respectively. The algorithm described here is derived from Section 5 of Kandel, et al. (1996).

Table 6

Vertices for second-order graph with three conditions

		Trial N-1	
Trial N-2	A	B	C
A	AA	AB	AC
B	BA	BB	BC
C	CA	CB	CC

Note: Vertices for the second-order graph comprise all of the ordered pairs of the three conditions.

Figure Captions

Figure 1. (A) Directional graph representing flights between airports. Each vertex represents an airport and edges between the vertices represent non-stop flights between them. Arrows indicate the direction of the flight. Weights on the edges represent flight durations. Recurrent edges (starting and ending at the same vertex) represent layover times. (B) A fully-connected, directional graph representing four experimental conditions, A-D, and the temporal relationships between them.

Figure 2. Demonstration of Hierholzer's algorithm to generate an Eulerian circuit through a graph with three vertices representing three conditions: A, B, C. The steps in the algorithm are shown separately. Gray dotted edges indicate edges in the graph which have been traversed in those steps and the accompanying numbers adjacent to each edge indicate the order in which the edges were traversed. The sequences, *T* and *T'*, are shown at each stage.

Figure 3. An example of a second-order counterbalancing graph representing two conditions.

Figure 4. An example of how seeing one condition may change a participant's strategy. (A) The sequence of letters "ANHUM" is hidden in the space between the black blocks. This makes it less likely to be noticed. (B) The sequence of letters "APEPL" is depicted in block letters and the letters are clearly noticeable. Once this stimulus is seen participants may change their strategy and look for letters in the other conditions.

Figure A1. Demonstration of how the number of Euler circuits of a graph is not equivalent to the number of condition sequences. (A) A graph with three vertices and edges between them. (B) Cyclic representation of an Euler circuit through the graph in panel A. The condition sequence generated from the Euler path depends on the starting vertex chosen. In this case, A in the upper left side is chosen and the condition sequence is shown below. (C) A different starting point from the same Euler circuit as in panel B results in a different sequence.

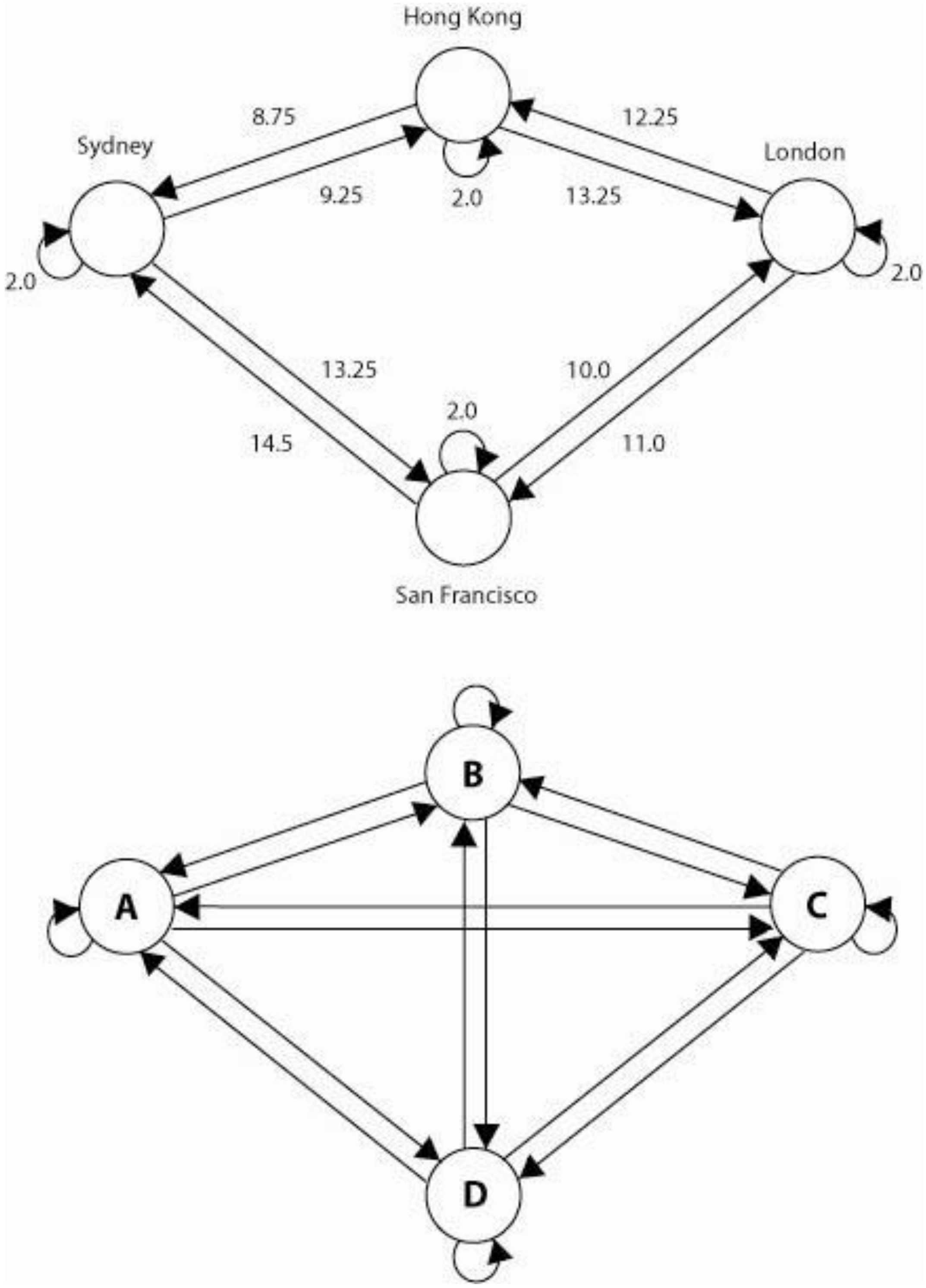
Hong Kong
8.75
12.25
Sydney
London
9.25
2.0
13.25
2.0
2.0
13.25
10.0
2.0
14.5
11.0
San Francisco
B
A
C
D

Figure 1

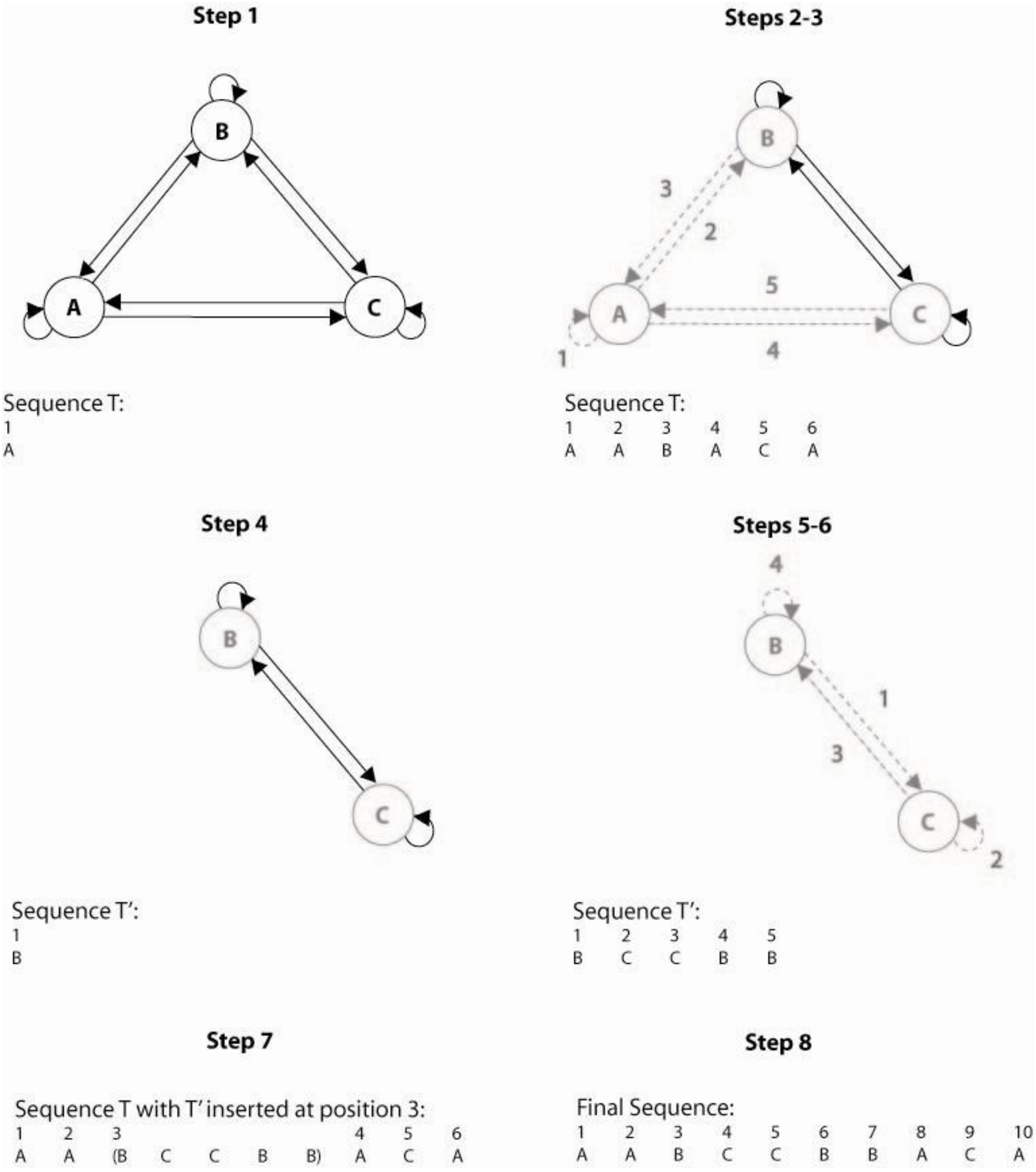
Step 1
B
A
C
Sequence T:
1
A
Steps 2-3
B
3
2
5
A
C
1
4
Sequence T:
1 2 3 4 5 6
A A B A C A
Step 4
B
C
Sequence T':
1
B
Steps 5-6
4
B
1
3
C
2
Sequence T':
1 2 3 4 5
B C C B B
Step 7
Sequence T with T' inserted at position 3:
1 2 3 4 5 6
A A (B C C B B) A C A
Step 8
Final Sequence:
1 2 3 4 5 6 7 8 9 10
A A B C C B B A C A

Figure 2

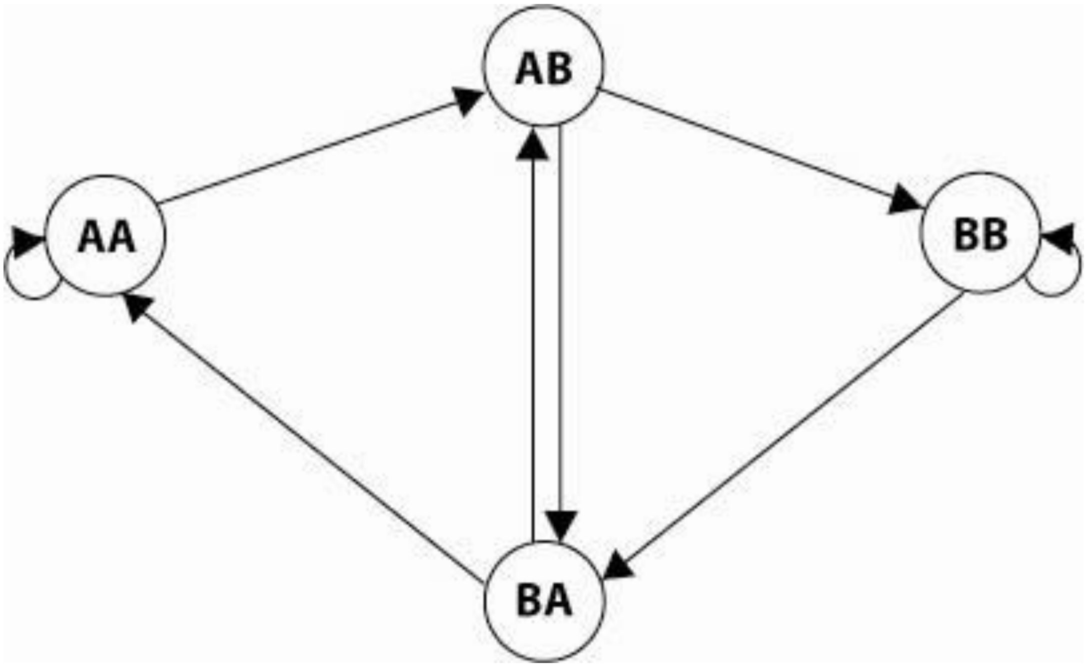

Figure 3

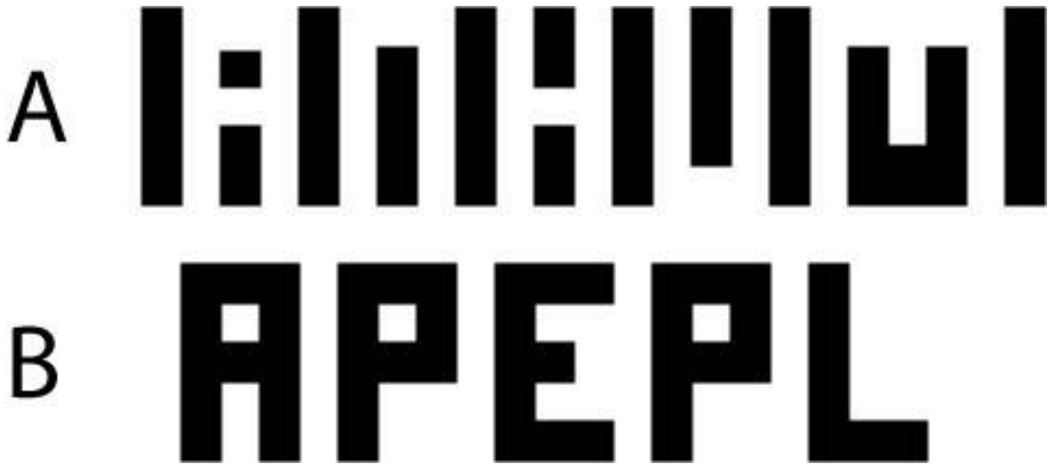

Figure 4

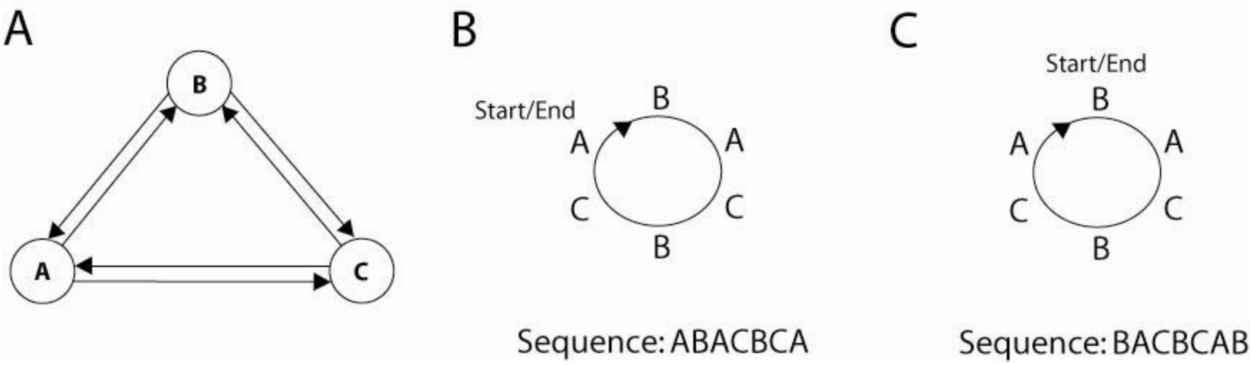
A
B
A
C
B
Start/End
B
A
A
C
C
B
Sequence: ABACBCA
C
Start/End
B
A
A
C
C
B
Sequence: BACBCAB

Figure A1