Formal Power Series and Algebraic Combinatorics
Nankai University, Tianjin, China, 2007

# Multiparking Functions, Graph Searching, and the Tutte Polynomial
# Extended Abstract

Dimitrije Kostić and Catherine Yan

Abstract. A parking function of length $n$ is a sequence $(b_1, b_2, \ldots, b_n)$ of nonnegative integers for which there is a permutation $\pi \in S_n$ so that $0 \leq b_{\pi(i)} < i$ for all $i$. A well-known result about parking functions is that the polynomial $P_n(q)$, which enumerates the complements of parking functions by the sum of their terms, is the generating function for the number of connected graphs by the number of excess edges when evaluated at $1+q$. In this paper we extend this result to arbitrary connected graphs $G$. In general the polynomial that encodes information about subgraphs of $G$ is the Tutte polynomial $t_G(x, y)$, which is the generating function for two parameters, namely the internal and external activities, associated with the spanning trees of $G$. We define $G$-multiparking functions, which generalize the $G$-parking functions that Postnikov and Shapiro introduced in the study of certain quotients of the polynomial ring. We construct a family of algorithmic bijections between the spanning forests of a graph $G$ and the $G$-multiparking functions. In particular, the bijection induced by the breadth-first search leads to a new characterization of external activity, and hence a representation of Tutte polynomial by the reversed sum of $G$-multiparking functions.

## 1. Introduction

The (classical) parking functions of length $n$ are sequences $(b_1, b_2, \ldots, b_n)$ of nonnegative integers for which there is a permutation $\pi \in S_n$ so that $0 \leq b_{\pi(i)} < i$ for all $i$. This notion was first introduced by Konheim and Weiss [**8**] in the study of the linear probes of random hashing function. Parking functions have many interesting combinatorial properties. The most notable one is that the number of parking functions of length $n$ is $(n+1)^{n-1}$, Cayley's formula for the number of labeled trees on $n+1$ vertices. This relation motivated much work in the early study of parking functions, in particular, combinatorial bijections between the set of parking functions of length $n$ and labeled trees on $n+1$ vertices.

There are a number of generalizations of parking functions, for example, see [**3**] for the double parking functions, [**16, 18, 19**] for $k$-parking functions, and [**13, 10**] for parking functions associated with an arbitrary vector. Recently, Postnikov and Shapiro [**14**] proposed a new generalization, the *$G$-parking functions*, associated to a general connected digraph $D$. Let $G$ be a digraph on $n+1$ vertices indexed by integers from 0 to $n$. A $G$-parking function is a function $f$ from $[n]$ to $\mathbb{N}$, the set of non-negative integers, satisfying the following condition: for each subset $U \subseteq [n]$ of vertices of $G$, there exists a vertex $j \in U$ such that the number of edges from $j$ to vertices outside $U$ is greater than $f(j)$. For the complete graph $G = K_{n+1}$, such defined functions are exactly the classical parking functions, where one views $K_{n+1}$ as the digraph with one directed edge $(i, j)$ for each pair $i \neq j$. In [**2**] Chebikin and Pylyavskyy constructed a family of bijections between the set of $G$-parking functions and the (oriented) spanning trees of that graph.

Perhaps the most important statistic of the classical parking functions is the (reversed) sum, that is, $\binom{n}{2} - (x_1 + x_2 + \cdots + x_n)$ for a parking function $(x_1, x_2, \ldots, x_n)$ of length $n$. It corresponds to the number of linear probes in hashing functions [**7**], the number of inversions in labeled trees on $[n+1]$ [**12**], and the number of hyperplanes separating a given region from the base region in the extended Shi arrangements

2000 *Mathematics Subject Classification.* Primary 05C30; Secondary 05C05.
*Key words and phrases.* parking functions, breadth-first search, Tutte polynomial, spanning forest.
The first author was partially supported by Texas A&M's NSF VIGRE grant.
The second author was partially support by NSF grant DMS 0245526.

[**16**], to list a few. It is also closely related to the number of connected graphs on $[n+1]$ with a fixed number of edges. In [**19**] the second author gave a combinatorial explanation, which revealed the underlying correspondence between the classical parking functions and labeled, connected graphs. The main idea is to use breadth-first search to find a labeled tree on any given connected graph, and record such a search by a queue process.

The objective of the present paper is to extend the result of [**19**] to arbitrary graphs. To describe properties of subgraphs for a general graph $G$, we use the Tutte polynomial, one of the most fundamental object in algebraic graph theory. An important approach to derive the Tutte polynomial is to use partitions, which dated back to the 1960's, see Crapo [**4**]. A nice illustration was given by Gessel and Sagan who used depth-first search to partition the substructures of a given graph by associating a spanning forest $F$ with each substructure. This process partitions the simplicial complex of all substructures (ordered by inclusion) into intervals, one for each $F$. Every interval turns out to be a Boolean algebra consisting of all ways to add external active edges to $F$. Expressing the Tutte polynomial in terms of sums over such intervals permits one to extract the necessary combinatorial information.

In [**6**] Gessel and Sagan also mentioned another search, the *neighbors-first search*, and related the external activity determined by the neighbors-first search on a complete graph with $n+1$ vertices to the sum of (classical) parking functions of length $n$. This connection was further explained in [**19**]. In the present paper we extend this result to an arbitrary graph $G$ by developing the connection between Tutte polynomial of $G$ and certain restricted functions defined on $V(G)$, the vertex set of $G$. This is achieved by combining the two approaches mentioned before. First, we use breadth-first search to get a new partition of all spanning subgraphs of $G$. Each subgraph is associated with a spanning forest of $G$, which allows us to get a new expression of the Tutte polynomial in terms of breadth-first external activities of its spanning forests. Second, we construct bijections between the set of all spanning forests of $G$ and the set of functions defined on $V(G)$ with certain restrictions. One of such bijection, namely the one induced by breadth-first search with a queue, leads to the characterization of the (breadth-first) external activity of a spanning forest by the corresponding function.

To work with spanning forests, we propose the notion of a $G$-multiparking function, a natural extension of the notion of a $G$-parking function. Let $G$ denote a graph with a totally ordered vertex set $V(G)$. Often we will take $V(G)=[n]=\{1,2,\ldots,n\}$. For simplicity and clarity, we assume that $G$ is a simple graph. For any subset $U\subseteq V(G)$, and vertex $v\in U$, define $outdeg_U(v)$ to be the cardinality of the set $\{\{v,w\}\in E(G)|w\notin U\}$. Here $E(G)$ is the set of edges of $G$.

Definition 1.1. Let $G$ be a simple graph with $V(G)=[n]$. A *$G$-multiparking function* is a function $f: V(G)=[n]\to\mathbb{N}\cup\{\infty\}$, such that for every $U\subseteq V(G)$ either **(A)** $i$ is the vertex of smallest index in $U$, (written as $i=\min(U)$), and $f(i)=\infty$, or **(B)** there exists a vertex $i\in U$ such that $0\le f(v_i)< outdeg_U(i)$.

The vertices which satisfy $f(i)=\infty$ in **(A)** will be called *roots of $f$* and those that satisfy **(B)** (in $U$) are said to be *well-behaved* in $U$, and **(A)** and **(B)** will be used to refer, respectively, to these conditions hereafter. We will describe the bijection between $G$-multiparking functions and the spanning forests in the next section, and reveal the relation between $G$-multiparking functions and the Tutte polynomial of $G$ in Section 3. Some properties of $G$-multiparking functions and enumerative results are given in Section 4, for which more details can be found in the full-length paper [**9**].

## 2. Bijections between multiparking functions and spanning forests

In this section, we construct bijections between the set $\mathcal{MP}_G$ of $G$-multiparking functions and the set $\mathcal{F}_G$ of spanning forests of $G$. A *sub-forest* $F$ of $G$ is a subgraph of $G$ without cycles. A leaf of $F$ is a vertex $v\in V(F)$ with degree 1 in $F$. Denote the set of leaves of $F$ by $Leaf(F)$. Let $\prod$ be the set of all ordered pairs $(F,W)$ such that $F$ is a sub-forest of $G$, and $\emptyset\neq W\subseteq Leaf(F)$. A *choice function* $\gamma$ is a function from $\prod$ to $V(G)$ such that $\gamma(F,W)\in W$.

Fix a choice function $\gamma$. Given a $G$-multiparking function $f\in\mathcal{MP}_G$, we define an algorithm to find a spanning forest $F\in\mathcal{F}_G$. Explicitly, we define quadruples $(val_i,P_i,Q_i,F_i)$ recursively for $i=0,1,\ldots,n$, where $val_i: V(G)\to\mathbb{Z}$ is the *value function*, $P_i$ is the set of *processed* vertices, $Q_i$ is the set of vertices *to be processed*, and $F_i$ is a subforest of $G$ with $V(F_i)=P_i\cup Q_i$, $Q_i\subseteq Leaf(F_i)$ or $Q_i$ consists of an isolated vertex of $F_i$.

**Algorithm A**.

- **Step 1: initial condition.** Let $val_0 = f$, $P_0$ be empty, and $F_0 = Q_0 = \{1\}$.
- **Step 2: choose a new vertex $v$.** At time $i \geq 1$, let $v = \gamma(F_{i-1}, Q_{i-1})$, where $\gamma$ is the choice function.
- **Step 3: process vertex $v$.** For every vertex $w$ adjacent to $v$ and $w \notin P_{i-1}$, set $val_i(w) = val_{i-1}(w) - 1$. For any other vertex $u$, set $val_i(u) = val_{i-1}(u)$. Let $N = \{w | val_i(w) = -1, val_{i-1}(w) \neq -1\}$. Update $P_i$, $Q_i$ and $F_i$ by letting $P_i = P_{i-1} \cup \{v\}$, $Q_i = Q_{i-1} \cup N \setminus \{v\}$ if $Q_{i-1} \cup N \setminus \{v\} \neq \emptyset$, otherwise $Q_i = \{u\}$ where $u$ is the vertex of the lowest-index in $[n] - P_i$. Let $F_i$ be a graph on $P_i \cup Q_i$ whose edges are obtained from those of $F_{i-1}$ by joining edges $\{w, v\}$ for each $w \in N$. We say that the vertex $v$ is processed at time $i$.

Iterate steps 2-3 until $i = n$. We must have $P_n = [n]$ and $Q_n = \emptyset$. Define $\Phi = \Phi_{\gamma,G} : \mathcal{MP}_G \to \mathcal{F}_G$ by letting $\Phi(f) = F_n$.

If an edge $\{v, w\}$ is added to the forest $F_i$ as described in Step 3, we say that *$w$ is found by $v$*, and $v$ is the *parent* of $w$, if $v \in P_{i-1}$. (In this paper, the parent of vertex $v$ will be frequently denoted $v^p$.) By Step 3, a vertex $w$ is in $Q_i$ because either it is found by some $v$ that has been processed, and $\{v, w\}$ is the only edge of $F_i$ that has $w$ as an endpoint, or $w$ is the lowest-index vertex in $[n] - P_i$ and is an isolated vertex of $F_i$. Also, it is clear that each $F_i$ is a forest, since every edge $\{u, w\}$ in $F_i \setminus F_{i-1}$ has one endpoint in $V(F_i) \setminus V(F_{i-1})$. Hence $\gamma(F_i, Q_i)$ is well-defined and thus we have a well-defined map $\Phi$ from $\mathcal{MP}_G$ to $\mathcal{F}_G$. The following proposition describes the role played by the roots of a $G$-multiparking function $f$.

Proposition 2.1. *Let $f$ be a $G$-multiparking function. Each tree component $T$ of $\Phi(f)$ has exactly one vertex $v$ with $f(v) = \infty$. In particular, $v$ is the least vertex of $T$.*

From the algorithm A we also see that the forest $F = \Phi(f)$ is built tree by tree. That is, if $T_i$ and $T_j$ are tree components of $F$ with roots $r_i$, $r_j$ and $r_i < r_j$, then every vertex of tree $T_i$ is processed before any vertex of $T_j$.

To show that $\Phi$ is a bijection, we define a new algorithm to find a $G$-multiparking function for any given spanning forest, and prove that it gives the inverse map of $\Phi$.

Let $G$ be a graph on $[n]$ with a spanning forest $F$. Let $T_1, \ldots, T_k$ be the trees of $F$ with respective minimal vertices $r_1 = 1 < r_2 < \cdots < r_k$.

**Algorithm B.**

- **Step 1. Determine the process order $\pi$.** Define a permutation $\pi = (\pi(1), \pi(2), \ldots, \pi(n)) = (v_1 v_2 \ldots v_n)$ on the vertices of $G$ as follows. First, $v_1 = 1$. Assuming $v_1, v_2, \ldots, v_i$ are determined. Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $W = \{v \notin V_i : v \text{ is adjacent to some vertices in } V_i\}$.
  - **Case (1)** If there is no edge of $F$ connecting vertices in $V_i$ to vertices outside $V_i$, let $v_{i+1}$ be the vertex of smallest index not already in $V_i$;
  - **Case (2)** Otherwise, let $F'$ be the forest obtained by restricting $F$ to $V_i \cup W$. Let $v_{i+1} = \gamma(F', W)$.

  (Hereafter, when discussing process orders, we will write $v_i$ as $\pi(i)$.)
- **Step 2. Define a $G$-multiparking function $f = f_F$.** Set $f(r_1) = f(r_2) = \cdots = f(r_k) = \infty$. For any other vertex $v$, let $r_v$ be the minimal vertex in the tree containing $v$, and $v, v^p, u_1, \ldots, u_t, r_v$ be the unique path from $v$ to $r_v$. Set $f(v)$ to be the cardinality of the set $\{v_j | \{v, v_j\} \in E(G),\ \pi^{-1}(v_j) < \pi^{-1}(v^p)\}$.

To verify that a function $f = f_F$ defined in this way is a $G$-multiparking function, we need the following lemma.

Lemma 2.1. *Let $f : V(G) \to \mathbb{N} \cup \{\infty\}$ be a function. If $v \in U \subseteq V(G)$ obeys property* **(A)** *or property* **(B)** *and $W$ is a subset of $U$ containing $v$, then $v$ obeys the same property in $W$.*

Proof. If $f(v) = \infty$ and $v$ is the smallest vertex in $U$, then it is still the smallest vertex in $W$. If $v$ is well-behaved in $U$, then $0 \leq f(v) < outdeg_U(v)$ and as $W \subseteq U$, we have $outdeg_U(v) \leq outdeg_W(v)$. Thus $v$ is well-behaved in $W$.

□

The *burning algorithm* was developed by Dhar [**5**] to determine if a function on the vertex set of a graph had a property called *recurrence*. An equivalent description for $G$-parking functions is given in [**2**]. Here we extend the burning algorithm to $G$-multiparking functions, and write it in a linear form.

PROPOSITION 2.2. *A vertex function is a $G$-multiparking function if and only if there exists an ordering $\pi(1), \pi(2), \ldots, \pi(n)$ of the vertices of a graph $G$ such that for every $j$, $\pi(j)$ satisfies either condition* **(A)** *or condition* **(B)** *in $U_j := \{\pi(j), \ldots, \pi(n)\}$.*

PROPOSITION 2.3. *The Algorithm B, when applied to a spanning forest of $G$, yields a $G$-multiparking function $f = f_F$.*

*Sketch of the proof.* Let $\pi$ be the permutation defined in Step 1 of Algorithm B. It can be shown by induction that the vertices can be thrown out in the order $\pi(1), \pi(2), \ldots, \pi(n)$. As $\pi(1) = 1$, the vertex $\pi(1)$ clearly can be thrown out. Suppose $\pi(1), \ldots, \pi(k-1)$ can be thrown out, and consider $\pi(k)$.

If $f(\pi(k)) = \infty$, by Case (1) of step 1, $\pi(k)$ is the smallest vertex not in $\{\pi(1), \ldots, \pi(k-1)\}$. Thus it can be thrown out.

If $f(\pi(k)) \neq \infty$, there is an edge of the forest $F$ connecting $\pi(k)$ to a vertex $w$ in $\{\pi(1), \ldots, \pi(k-1)\}$. Suppose $w = \pi(t)$ where $t < k$. By definition of $f$, there are exactly $f(\pi(k))$ edges connecting $\pi(k)$ to the set $\{\pi(1), \ldots, \pi(t-1)\}$. Hence $f(\pi(k)) < outdeg_{\{\pi(k),\ldots,\pi(n)\}}(\pi(k))$. Thus $\pi(k)$ can be thrown out as well. □

Define $\Psi_{\gamma,G} : \mathcal{F}_G \to \mathcal{MP}_G$ by letting $\Psi_{\gamma,G}(F) = f_F$. Then

THEOREM 2.2. *$\Psi(\Phi(f)) = f$ for any $f \in \mathcal{MP}_G$ and $\Phi(\Psi(F)) = F$ for any $F \in \mathcal{F}_G$, where $\Phi = \Phi_{\gamma,G}$ and $\Psi = \Psi_{\gamma,G}$.*

Since the roots of the $G$-multiparking function correspond exactly to the minimal vertices in the tree components of the corresponding forest, in the following we will refer to those vertices as *roots of the forest.*

## 3. External activity and the Tutte polynomial

**3.1. $F$-redundant edges.** A forest $F$ on $[n]$ may appear as a subgraph of different graphs, and a vertex function $f$ may be a $G$-multiparking function for different graphs. In this section we characterize the set of graphs which share the same pair $(F, f)$. Again let $G$ be a simple graph on $[n]$, and fix a choice function $\gamma$. For a spanning forest $F$ of $G$, let $f = \Psi_{\gamma,G}(F)$. We say an edge $e$ of $G - F$ is *$F$-redundant* if $\Psi_{\gamma,G-\{e\}}(F) = f$. Note that we only need to use the value of $\gamma$ on $(F', W)$ where $F'$ is a sub-forest of $F$. Hence $\Psi_{\gamma,G-\{e\}}(F)$ is well-defined.

Let $\pi$ be the order defined in Step 1 of Algorithm B. Note that $\pi$ only depends on $F$, not the underlying graph $G$. Recall that $v^p$ denotes the parent vertex of vertex $v$ in some spanning forest. We have the following proposition.

PROPOSITION 3.1.
*An edge $e = \{v, w\}$ of $G$ is $F$-redundant if and only if $e$ is one of the following types:*

1. *Both $v$ and $w$ are roots of $F$.*
2. *$v$ is a root and $w$ is a non-root of $F$, and $\pi^{-1}(w) < \pi^{-1}(v)$.*
3. *$v$ and $w$ are non-roots and $\pi^{-1}(v^p) < \pi^{-1}(w) < \pi^{-1}(v)$. In this case $v$ and $w$ must lie in the same tree of $F$.*

PROOF. We first show that each edge of the above three types are $F$-redundant. Since for any root $r$ of the forest $F$, $f(r) = \infty$, the edges of the first two types play no role in defining the function $f$. And clearly those edges are not in $F$. Hence they are $F$-redundant.

For edge $(v, w)$ of type 3, clearly it cannot be an edge of $F$. Since $f(v) = \#\{v_j | (v, v_j) \in E(G), \pi^{-1}(v_j) < \pi^{-1}(v^p)\}$, and $\pi^{-1}(w) > \pi^{-1}(v^p)$, removing the edge $\{v, w\}$ would not change the value of $f(v)$. This edges has no contribution in defining $f(u)$ for any other vertex $u$. Hence it is $F$-redundant.

For the converse, suppose that $e = \{v, w\}$ is not one of the three type. Assume $w$ is processed before $v$ in $\pi$. Then $v$ is not a root, and $w$ appears before $v^p$. Then removing the edge $e$ will change the value of $f(v)$. Hence it is not $F$-redundant.

□

Let $R_1(G; F)$, $R_2(G; F)$, and $R_3(G; F)$ denote the sets of $F$-redundant edges of types 1, 2, and 3, respectively. Among them, $R_3(G; F)$ is the most interesting one, as $R_1(G; F)$ and $R_2(G; F)$ are a consequence of the requirement that $f(r) = \infty$ for any root $r$. Let $R(G; F)$ be the union of these three sets. Clearly the $F$-redundant edges are mutually independent, and can be removed one by one without changing the corresponding $G$-multiparking function. Hence

Theorem 3.1. *Let $H$ be a subgraph of $G$ with $V(H) = V(G)$. Then $\Psi_{\gamma,G}(F) = \Psi_{\gamma,H}(F)$ if and only if $G - R(G;F) \subseteq H \subseteq G$.*

**3.2. A classification of the edges of $G$.** The notion of $F$-redundancy allows us to classify the edges of a graph in terms of the algorithm A. Roughly speaking, the edges of any graph can be thought of as either lowering $val(v)$ for some $v$ to 0, being in the forest, or being $F$-redundant. Explicitly, we have

Proposition 3.2.
*Let $f$ be a $G$-multiparking function and let $F = \Phi(f)$. Then*

$$|E(G)| = \left( \sum_{v: f(v) \neq \infty} f(v) \right) + |E(F)| + |R(G;F)|.$$

Proof. For each non-root vertex $v$, the number of different values that $val_i(v)$ takes on during the execution of algorithm $A$ is $f(v) + 1 + n_v$, where $n_v = -val_n(v)$. At the beginning, $val_0(v) = f(v)$. The value $val_i(v)$ then is lowered by one whenever there is a vertex $w$ which is adjacent to $v$ and processed before $v^p$. When $v^p$ is being processed, $val_i(v) = -1$, and the edge $\{v^p, v\}$ contributes to the forest $F$. Afterward, the value of $val_i(v)$ decreases by 1 for each $F$-redundant edge $\{u, v\}$ with $\pi^{-1}(u) < \pi^{-1}(v)$. Summing over all non-root vertices gives

$$\sum_{v: f(v) \neq \infty} deg_{<_\pi}(v) \quad = \quad \sum_{v: f(v) \neq \infty} f(v) + |E(F)| + \sum_{v: f(v) \neq \infty} n_v,$$

where $deg_{<_\pi}(v) = |\{\{w, v\} \in E(G) | \pi^{-1}(w) < \pi^{-1}(v)\}|$.

The edges that lower $val(v)$ below $-1$ are exactly the $F$-redundant edges of type (3) in Prop. 3.1, hence $\sum_{v: f(v) \neq \infty} n_v = |R_3(G;F)|$. On the other hand, $\sum_{v: f(v) \neq \infty} deg_{<_\pi}(v)$ is exactly $|E(G)| - |R_1(G;F)| - |R_2(G;F)|$. The claim follows from the fact that the sets $R_1(G;F), R_2(G;F)$, and $R_3(G;F)$ are mutually exclusive.

□

One notes that for roots of $f$ and $F = \Phi(f)$, $|R_1(G;F)| + |R_2(G;F)|$ is exactly $\sum_{root\ v} deg_{<_\pi}(v)$, where $\pi$ is the processing order in algorithm A. But it is not necessary to run the full algorithm A to compute $|R_1(G;F)| + |R_2(G;F)|$. Instead, we can apply the burning algorithm in a greedy way to find an ordering $\pi' = v_1' v_2' \cdots v_n'$ on $V(G)$: Let $v_1' = 1$. After determining $v_1', \dots, v_{i-1}'$, if $V_i = V(G) - \{v_1', \dots, v_{i-1}'\}$ has a well-behaved vertex, let $v_i'$ be one of them; otherwise, let $v_i'$ be the minimal vertex of $V_i$, (which has to be a root.)

$\pi'$ may not be the same as $\pi$, but they have the following properties:

1. Let $r_1 < r_2 < \cdots < r_k$ be the roots of $f$. Then $r_1, r_2, \dots, r_k$ appear in the same positions in both $\pi$ and $\pi'$.
2. The set of vertices lying between $r_i$ and $r_{i+1}$ are the same in $\pi$ and $\pi'$. In fact, they are the vertices of the tree $T_i$ with root $r_i$ in $F = \Phi(f)$.

It follows that for any root vertex $v$, $deg_{<_\pi}(v) = deg_{<_{\pi'}}(v)$. We call $deg_{<_\pi}(v)$ the *record* of the root $v$, and denote it by $rec(v)$. Then

$$|R_1(G;F)| + |R_2(G;F)| = \sum_{root\ v} deg_{<_{\pi'}}(v) = \sum_{root\ v} rec(v)$$

is the *total root records*. Let $Rec(f) = |R_1(G;F)| + |R_2(G;F)|$. It is the number of $F$-redundant edges adjacent to a root. By the above greedy burning algorithm, the total root records $Rec(f)$ can be computed in linear time.

**3.3. A new expression for Tutte polynomial.** In this subsection we relate $G$-multiparking functions to the Tutte polynomial $t_G(x, y)$ of $G$. We follow the presentation of [**6**]. Suppose we are given $G$ and a total ordering of its edges. Consider a spanning tree $T$ of $G$. An edge $e \in G - T$ is *externally active* if it is the largest edge in the unique cycle contained in $T \cup e$. We let

$$\mathcal{EA}(T) = \text{set of externally active edges in } T$$

and $ea(T) = |\mathcal{EA}(T)|$. An edge $e \in T$ is *internally active* if it is the largest edge in the unique cocycle contained in $(G-T) \cup e$ (a *cocycle* is a minimal edge cut). We let

$$\mathcal{IA}(T) = \text{set of internally active edges in } T$$

and $ia(T) = |\mathcal{IA}(T)|$. Tutte [**17**] then defined his polynomial as

$$t_G(x,y) = \sum_{T \subset G} x^{ia(T)} y^{ea(T)}, \tag{3.1}$$

where the sum is over all spanning trees $T$ of $G$. Tutte showed that $t_G$ is well-defined, i.e., independent of the total ordering of the edges of $G$. Henceforth, we will not assume that the edges of $G$ are ordered.

Let $H$ be a (spanning) subgraph of $G$. Denote by $c(H)$ the number of components of $H$. Define two invariants associated with $H$ as

$$\sigma(H) = c(H) - 1, \qquad \sigma^*(H) = |E(H)| - |V(G)| + c(H). \tag{3.2}$$

The following identity is well-known, for example, see [**1**].

Theorem 3.2.

$$t_G(1+x, 1+y) = \sum_{H \subseteq G} x^{\sigma(H)} y^{\sigma^*(H)}, \tag{3.3}$$

*where the sum is over all spanning subgraphs $H$ of $G$.*

Recall that the *breadth-first search* (BFS) is an algorithm that gives a spanning forest in the graph $H$. Assume $V(G) = [n]$. We will use our favorite description to express the BFS as a queue $Q$ that starts at the least vertex 1. This description was first introduced in [**15**] to develop an exact formula for the number of labeled connected graphs on $[n]$ with a fixed number of edges, and was used by the second author in [**19**] to reveal the connection between the classical parking functions (resp. $k$-parking functions) and the complete graph (resp. multicolored graphs).

Given a subgraph $H$ of $G$ with $V(H) = V(G) = [n]$, we construct a queue $Q$. At time 0, $Q$ contains only the vertex 1. At each stage we take the vertex $x$ at the head of the queue, remove $x$ from the queue, and add all unvisited neighbors $u_1, \ldots, u_{t_x}$ of $x$ to the queue, in numerical order. We will call this operation "processing $x$". If the queue becomes empty, add the least unvisited vertex to $Q$. The output $F$ is the forest whose edge set consists of all edges of the form $\{x, u_i\}$ for $i = 1, \ldots t_x$. We will denote this output as $F = BFS(H)$. Figure 1 shows the spanning forest found by BFS for a graph $G$.

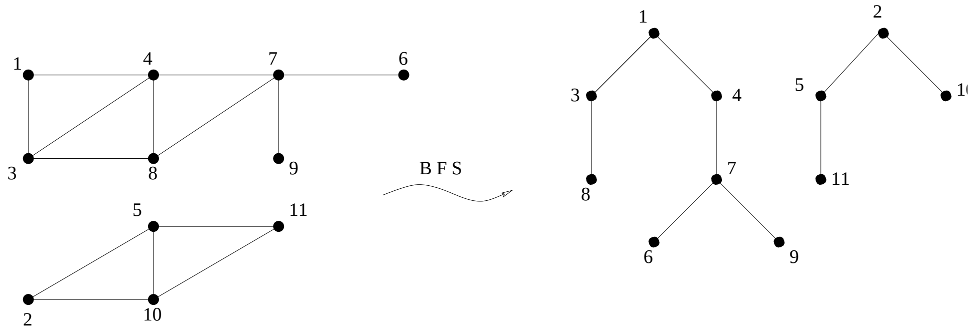

Figure 1. Spanning forest found by BFS.

For a spanning forest $F$ of $G$, let us say that an edge $e \in G - F$ is *BFS-externally active* if $BFS(F \cup e) = F$. A crucial observation is made by Spencer [**15**]: An edge $\{v, w\}$ can be added to $F$ without changing the spanning forest under the BFS if and only if the two vertices $v$ and $w$ have been present in the queue at the same time. In our example of Figure 1, edges $\{3,4\}, \{4,8\}, \{7,8\}, \{6,9\}, \{5,10\}$ and $\{10,11\}$ could be added back to $F$. We write $\mathcal{E}(F)$ for the set of BFS-externally active edges.

Proposition 3.3 (Spencer). *If $H$ is any subgraph and $F$ is any spanning forest of $G$ then*

$$BFS(H) = F \textit{ if and only if } F \subseteq H \subseteq F \cup \mathcal{E}(F).$$

Now consider the Tutte polynomial. Note that if $BFS(H) = F$, then $c(H) = c(F)$. So $\sigma(H) = c(F) - 1$ and $\sigma^*(H) = |E(H)| - |E(F)| = |\mathcal{E}(F) \cap H|$. Hence if we fix a forest $F$ and sum over the corresponding interval $[F, F \cup \mathcal{E}(F)]$, we have

$$\sum_{H:BFS(H)=F} x^{\sigma(H)} y^{\sigma^*(H)} = x^{c(F)-1} \sum_{A \subseteq \mathcal{E}(F)} y^{|A|} = x^{c(F)-1}(1+y)^{|\mathcal{E}(F)|}.$$

Summing over all forests $F$, we get

$$t_G(1+x, 1+y) = \sum_{H \subseteq G} x^{\sigma(H)} y^{\sigma^*(H)} = \sum_{F \subseteq G} x^{c(F)-1}(1+y)^{|\mathcal{E}(F)|}.$$

Or, equivalently,

$$t_G(1+x, y) = \sum_{F \subseteq G} x^{c(F)-1} y^{|\mathcal{E}(F)|}. \tag{3.4}$$

To evaluate $\mathcal{E}(F)$, note that when applying BFS to a graph $H$, the queue $Q$ only depends on the spanning forest $F = BFS(H)$. Given a forest $F$, the processing order in $Q$ is a total order $<_Q = <_Q (F)$ on the vertices of $F$ satisfying the following condition: Let $T_1, T_2, \ldots, T_k$ be the tree components of $F$ with minimal elements $r_1 = 1 < r_2 < \cdots < r_k$. Then (1) If $v$ is a vertex in tree $T_i$, $w$ is a vertex in tree $T_j$ and $i < j$, then $v <_Q w$. (2) Among vertices of each tree $T_i$, $r_i$ is minimal in the order $<_Q$. (3) For two non-root vertices $v, w$ in the same tree, $v <_Q w$ if $v^p <_Q w^p$. In the case $v^p = w^p$, $v <_Q w$ whenever $v < w$. Note that the condition that two vertices $v, w$ have been present at the queue $Q$ at the same time when applying BFS to $F$ is equivalent to $v^p <_Q w <_Q v$ or $w^p <_Q v <_Q w$. That is, an edge is BFS-externally active if and only if it is an $F$-redundant edge of type 3, as defined in §3.1, in which the choice function $\gamma(F, W) = v$ where $v$ is the minimal element in $W$ under the breadth-first search order on $F$. It follows that $\mathcal{E}(F) = R_3(G; F)$.

Therefore by Prop. 3.2,

$$|\mathcal{E}(F)| = |R_3(G; F)| = |E(G)| - |E(F)| - \left( \sum_{v: f(v) = \infty} f(v) \right) - Rec(f),$$

where $f = \Psi_{\gamma,G}(F)$ is the corresponding $G$-multiparking function. Note that $|E(F)| = n - c(F)$, and $c(F) = r(f)$, where $r(f)$ is the number of roots of $f$. Therefore

Theorem 3.3.

$$t_G(1+x, y) = y^{|E(G)|-n} \sum_f x^{r(f)-1} y^{r(f) - Rec(f) - \left(\sum_{v: f(v) \neq \infty} f(v)\right)},$$

*where the sum is over all $G$-multiparking functions.*

For a $G$-multiparking function $f$, where $G$ is a graph on $n$ vertices, we call the statistics $|E(G)| - n + r(f) - Rec(f) - \sum_{v: f(v) \neq \infty} f(v)$ the *reversed sum* of $f$, denote by $rsum(f)$. The name comes from the corresponding notation for classical parking functions, see, for example, [**11**]. Theorem 3.3 expresses Tutte polynomial in terms of generating functions of $r(f)$ and $rsum(f)$. In [**6**] Gessel and Sagan gave a similar expression, in terms of $\mathcal{E}_{DFS}(F)$, the set of *greatest-neighbor externally active* edges of $F$, which is defined by applying the greatest-neighbor depth-first search on subgraphs of $G$. Combining the result of [**6**] (Formula 5), we have

$$x t_G(1+x, y) = \sum_{F \subseteq G} x^{c(F)} y^{|\mathcal{E}_{DFS}(F)|} = \sum_{F \subseteq G} x^{c(F)} y^{|\mathcal{E}(F)|} = \sum_{f \in \mathcal{MP}_G} x^{r(f)} y^{rsum(f)}.$$

That is, the three pairs of statistics, $(c(F), |\mathcal{E}_{DFS}(F)|)$ and $(c(F), |\mathcal{E}(F)|)$ for spanning forests, and $(r(f), rsum(f))$ for $G$-multiparking functions, are all equally distributed.

## 4. Enumeration of $G$-multiparking functions and graphs

In this section we discuss some enumerative results on $G$-multiparking functions and substructures of graphs.

Theorem 4.1. *The number of $G$-multiparking functions with $k$ roots equals the number of spanning forests of $G$ with $k$ components. In particular, for connected graph $G$, the number of $G$-multiparking functions is $T_G(2,1)$. Among them, those with an odd number of roots is counted by $\frac{1}{2}(T_G(2,1)+T_G(0,1))$, and those with an even number of roots is counted by $\frac{1}{2}(T_G(2,1)-T_G(0,1))$.*

Proof. The first two sentences follow directly from the bijections constructed in §2, and Theorem 3.3. For the third sentence, just note that $T_G(0,1)=\sum_f(-1)^{r(f)-1}$ is the difference between the number of $G$-multiparking functions with an odd number of roots, and those with an even number of roots. □

Corollary 4.2. *Let $G$ be a connected graph. The number $\gamma_{t,k}(G)$ of spanning subgraphs $H$ with $t$ components and $V(G)-1+k$ edges is given by*

$$\gamma_{t,k}(G)=\sum_{F\in\mathcal{F}_t}\binom{\mathcal{E}(F)}{k}=\sum_{f\in\mathcal{MP}_t(G)}\binom{rsum(f)}{k},$$

*where the first sum is over all spanning forests with $t$ components, and the second sum is over all $G$-multiparking functions with $t$ roots.*

Proof. For any spanning forest $F$ with $k$ components, the number of spanning subgraphs $H$ with $V(G)-1+k$ edges such that $BFS(H)=F$ is given by $\binom{\mathcal{E}(F)}{k}$. □

Next we give a new expression of the $t_{K_{n+1}}(x,y)$ in terms of classical parking functions. It enumerates the classical parking functions by the number of *critical left-to-right maxima*. Given a classical parking function $\mathbf{b}=(b_1,\ldots,b_n)$, we say that a term $b_i=j$ is *critical* if in $\mathbf{b}$ there are exactly $j$ terms less than $j$, and exactly $n-1-j$ terms larger than $j$. For example, in $\mathbf{b}=(3,0,0,2)$, the terms $b_1=3$ and $b_4=2$ are critical. Among them, only $b_1=3$ is also a left-to-right maximum.

Let $\alpha(\mathbf{b})$ be the number of critical left-to-right maxima in a classical parking function $\mathbf{b}$. We have

Theorem 4.3.

$$t_{K_{n+1}}(x,y)=\sum_{\mathbf{b}\in P_n}x^{\alpha(\mathbf{b})}y^{\binom{n}{2}-\sum_i b_i},$$

*where $P_n$ is the set of classical parking functions of length $n$.*

Proof. Let $F$ be a spanning forest on $[n+1]$ with tree components $T_1,\ldots,T_k$, where $T_i$ has minimal vertex $r_i$, and $r_1<r_2<\cdots<r_k$. We define an operation $merge(F)$ which combines the trees $T_1,\ldots,T_k$ by adding an edge between $r_i$ with $w_{i-1}$ for each $i=2,...,k$, where $w_{i-1}$ is the vertex of $T_{i-1}$ that is maximal under the order $<_{bf,q}$. Denote by $T_F=merge(F)$ the resulting tree. We observe that for the forest $F$ and the tree $T_F$, the queue obtained by applying BFS are exactly the same. This implies that $F$ and $T_F$ have the same set of BFS-externally active edges.

Conversely, given $T$ and an edge $e=\{w,v\}\in T$ where $w<_{bf,q}v$. We say the edge $e$ is *critical* in $T$ if $merge(T\setminus\{e\})=T$. Assume $T\setminus\{e\}=T_1\cup T_2$ where $w\in T_1$ and $v\in T_2$. By the definition of the merge operation, $e$ is critical if and only if $w$ is the maximal in $T_1$ under the order $<_{bf,q}$, and $v$ is vertex of the lowest index in $T_2$. In terms of the queue obtained by applying BFS to $T$, it is equivalent to the following two conditions: (1) There is an index $i$ such that $Q_i=\{v\}$, and $v$ does not belong to any other $Q_j$. (2) $v$ is of minimal index among the set of vertices processed after $v$.

Consider the maps $\Phi_{\gamma,G}$ and $\Psi_{\gamma,G}$ with $\gamma=\gamma_{bf,q}$ and $G=K_{n+1}$. Let $f=\Psi_{\gamma,G}(T)$, and write $f$ as a sequence $(f(2),f(3),\ldots,f(n+1))$. (There is no need to record $f(1)$, as $f(1)=\infty$ always.) Then an edge $\{w,v\}$ is critical in $T$ if and only if (1) $f(v)$ is critical in the sequence $(f(2),\ldots,f(n+1))$, and (2) $w>v$ for any vertex $w$ with $f(w)>f(v)$. That is, $f(v)$ is a left-to-right maximum in the sequence $(f(2),f(3),\ldots,f(n+1))$.

Now fix a spanning tree $T$ of $K_{n+1}$ and let $Merge(T)$ be the set of spanning forests $F$ such that $merge(F)=T$. Then an $F\in Merge(T)$ can be obtained from $T$ by removing any subset $A$ of critical edges, in which case $c(F)=c(T)+|A|=1+|A|$. This, combined with the fact that $\mathcal{E}(F)=\mathcal{E}(T)$, gives us

$$\sum_{F\in Merge(T)}x^{c(F)-1}y^{|\mathcal{E}(F)|}=y^{|\mathcal{E}(T)|}\sum_{A}x^{|A|}, \tag{4.1}$$

where $A$ ranges over all subsets of critical edges of $T$. Under the correspondence $T \to f = \Psi_{\gamma,G}(T)$ and considering $f$ as a sequence $(f(2), \dots, f(n+1))$, $|\mathcal{E}(T)|$ is just $\binom{n}{2} - \sum_{i=2}^{n+1} f(i)$, and critical edges of $T$ correspond to critical left-to-right maxima of the sequence. Hence the sum in (4.1) equals

$$y^{|\mathcal{E}(T)|}(1+x)^{\alpha(f_T)} = (1+x)^{\alpha(f_T)} y^{\binom{n}{2} - \sum_{i=2}^{n+1} f(i)}.$$

Theorem 4.3 follows by summing over all trees on $[n+1]$. □

Finally, we use the breadth-first search to re-derive the formula for the number of subdigraphs of $G$, which was first proved in [**6**] using DFS, and extend the method to derive a formula for the number of subtraffics of $G$.

Let $G$ be a graph. A *directed subgraph* or *subdigraph* of $G$ is a digraph $D$ that contains up to one copy of each orientation of every edge of $G$. Here for an edge $\{u, v\}$ of $G$ we permit both $(u, v)$ and $(v, u)$ to appear in a subdigraph.

For any subdigraph $D$ of $G$, we apply the BFS to get a spanning forest of $D$. The only difference from the subgraph case is that when processing a vertex $x$, we only add those unvisited vertices $u$ such $(x, u)$ is an edge of $D$.

If digraph $D$ has BFS forest $F$, write $\vec{\mathcal{F}}^+(D) = F$. Note that we can view $F$ as an oriented spanning forest, where each edge is pointing away from the root (i.e., the minimal vertex) of the underlying tree component. Say a directed edge $\vec{e} \notin F$ is *directed BFS externally active* with respect to $F$ if $\vec{\mathcal{F}}^+(F \cup \vec{e}) = F$. Denote by $\mathcal{E}^+(F)$ the set of directed BFS-externally active edges. Then we have the following basic proposition, which is the analog in the undirected case.

Proposition 4.1. *If $D$ is any subdigraph and $F$ is any spanning forest of $G$ then*

$$\vec{\mathcal{F}}^+(D) = F \text{ if and only if } F \subseteq D \subseteq F \cup \mathcal{E}^+(F).$$

Now we characterize the directed BFS-externally active edges by the set $\mathcal{E}(F)$, the BFS-externally active edges for the undirected graph $G$. Let $\{u, v\}$ be an edge of $G$ with $u <_{bf,q} v$. If $\{u, v\} \in E(F)$, then the backward edge $(v, u)$ can be added without changing the result of (directed) breadth-first search, that is, $(v, u) \in \mathcal{E}^+(F)$. If $\{u, v\} \in \mathcal{E}(F)$, then both $(u, v)$ and $(v, u)$ are in $\mathcal{E}^+(F)$. If $\{u, v\}$ is not in the forest $F$ or $\mathcal{E}(F)$, then $(v, u)$ is in $\mathcal{E}^+(F)$. Together we have

$$|E(G)| = |\mathcal{E}^+(F)| - |\mathcal{E}(F)|.$$

It is easy to compute that

Theorem 4.4. *If $G$ has $n$ vertices, then*

$$\sum_D x^{c(D)} y^{|E(D)|} = xy^{n-1}(1+y)^{|E(G)|}\ t_G(1 + \frac{x}{y}, 1+y), \tag{4.2}$$

*where the sum is over all subdigraphs of $G$.*

We extend the above method to a slightly complicated problem. The *sub-traffic* $K$ of $G$, where $K$ is a partially directed graph on $V(G)$, is obtained from $G$ by replacing each edge $\{u, v\}$ of $G$ by (a) $\emptyset$, (b) a directed edge $(u, v)$, (c) a directed edge $(v, u)$, (d) two directed edges $(u, v)$ and $(v, u)$, or (e) an undirected edge $\{u, v\}$. Using a similar argument as in counting the subdigraphs, we get

Theorem 4.5. *Let $G$ be a connected graph. Then*

$$\sum_K x^{c(K)} y^{|E(K)|} = x(y^2 + 2y)^{n-1}(1+y)^{|E(G)|-n+1}\ t_G(1 + \frac{x(1+y)}{y(2+y)}, \frac{1+3y+y^2}{1+y}), \tag{4.3}$$

*where the sum is over all subtraffic of $G$.*

It is easy to see that the number of subtraffics on $G$ is $5^{|E(G)|}$. Using this fact and evaluating equation 4.3 at $x = y = 1$, we derive the equality $3^{n-1}2^{|E(G)|-n+1}t_G(\frac{5}{3}, \frac{5}{2}) = 5^{|E(G)|}$. Evaluating the equation at $x = 0, y = 1$ proves that the number of connected subtraffics on $G$ is $3^{n-1}2^{|E(G)|-n+1}t_G(1, \frac{5}{2})$.

## Acknowledgments

The authors thank Robert Ellis and Jeremy Martin for helpful discussions and comments. We also thank Ira Gessel for helpful comments on Tutte polynomials, and for sharing the unpublished portion of a preprint of [**6**] with us.

## References

[1] N. Biggs, *Algebraic Graph Theory.* $2^{nd}$ ed, Cambridge University Press, 1993.

[2] D. Chebikin and P. Pylyavskyy, *A Family of Bijections Between G-Parking Functions and Spanning Trees.* Journal of Combinatorial Theory A **110** (2005), no. 1, 31-41.

[3] R. Cori and D. Poulalhon, *Enumeration of $(p, q)$-Parking Functions.* Discrete Math **256** (2002) 609-623.

[4] H.H. Crapo, The Tutte polynomial, *Aequationes Math.* **3**(1969), 211–229.

[5] D. Dhar, *Self-organized critical state of the sandpile automaton models*, Physical Review Letters **64**(1990), no. 14, 1613–1616.

[6] I. Gessel and B. Sagan, *The Tutte Polynomial of a Graph, Depth-First Search, and Simplicial Complex Partitions.* Electronic Journal of Combinatorics **3** (no. 2) R9.

[7] D. Knuth, *Linear probing and graphs, average-case analysis for algorithms.* Algorithmica **22**(1998), No.4, 561–568.

[8] A. G. Konheim and B. Weiss, *An Occupancy Discipline and Applications.* SIAM Journal of Applied Mathematics **14** (1966) 1266–1274.

[9] D. Kostic and C. H. Yan, *Multiparking Functions, Graph Searching, and the Tutte Polynomial*, preprint, available at `http://www.math.tamu.edu/∼cyan/papers.html`.

[10] J. P. Kung and C. H. Yan, *Gončarov polynomials and parking functions.* Journal of Combinatorial Theory, Series A, **102**(2003), 16–37.

[11] J. P. Kung and C. H. Yan, *Exact formula for moments of sums of classical parking functions*, Advances in Applied Mathematics, vol 31(2003), 215–241.

[12] G. Kreweras, *Une famille de polynômes ayant plusieurs propriétés énumeratives*, Period. Math. Hungar. **11**(1980), 309–320.

[13] J. Pitman and R. P. Stanley, *A polytope related to empirical distributions, plane trees, parking functions, and the associahedron.* Discrete and Computational Geometry, **27**(2002), no.4, 603–634.

[14] A. Postnikov and B. Shapiro, *Trees, Parking Functions, Syzygies, and Deformations of Monomial Ideals.* Transactions of the American Mathematical Society **356** (2004).

[15] J. Spencer, *Enumerating graphs and Brownian Motion*, Communications on Pure and Applied Mathematics, Vol. L (1997) 291–294.

[16] R. P. Stanley, *Hyperplane arrangements, parking functions, and tree inversions*, in B. Sagan and R. Stanley, eds., "Mathematical essays in honor of Gian-Carlo Rota," Birkhäuser, Boston and Basel, 1998, pp. 359–375.

[17] W. Tutte, *A Contribution to the Theory of Chromatic Polynomials.* Canadian Journal of Mathematics **6** (1953). 80-91.

[18] C. H. Yan, *Generalized tree inversions and k-parking functions.* Journal of Combinatorial Theory, Series A, **79**(1997), 268–280.

[19] C. H. Yan, *Generalized Parking Functions, Tree Inversions, and Multicolored Graphs.* Advances in Applied Mathematics **27** (2001). 641-670.

Department of Mathematics, Texas A&M University, College Station, Texas 77843. USA
*E-mail address*: `dkostic@math.tamu.edu`

Department of Mathematics, Texas A&M University, College Station, Texas 77843. USA
*E-mail address*: `cyan@math.tamu.edu`
*URL*: `http://www.math.tamu.edu/∼cyan`